Herold, Andreas

"Erstellung eines Frameworks für ein Communitynetzwerk"

eingereicht als

DIPLOMARBEIT

an der

HOCHSCHULE MITTWEIDA (FH)

UNIVERSITY OF APPLIED SCIENCES

Informationstechnik & Elektrotechnik

Dresden, 2009

Erstprüfer: Herr Prof. Dr.-Ing. Frank Zimmer
Zweitprüfer: Herr Dirk Oschmann

vorgelegte Arbeit wurde verteidigt am:

Bibliographische Beschreibung:

Herold, Andreas:
Erstellung eines Frameworks für ein Communitynetzwerk - 2009 - 80 S.
Mittweida, Hochschule Mittweida (FH), Fachbereich Informationstechnik & Elektrotechnik,
Diplomarbeit, 2009

Referat:
Ziel der vorliegenden Diplomarbeit ist es, ein funktionsfähiges Framework nach den Vorgaben des Auftraggebers zu konzipieren und zu erstellen. Die Arbeit stellt dabei vor allem den grundlegenden Prozess dar, der von der Aufgabenstellung beginnend über die Planung und Konzeption bis hin zur Umsetzung durchlaufen wird. Die Vorgehensweise sieht dabei so aus, dass das Problem als erstes dargestellt wird um im Anschluss Lösungen zu erstellen und zu diskutieren. In diesem Schritt wird unter anderen neben der Datenbank auch das Entwurfsmuster besprochen. Als letzter Schritt und zur Lösung des Problems wird die Umsetzung dargestellt. Besonderer Schwerpunkt liegt dabei vor allem bei der Erweiterbarkeit des Framework, so dass dieses flexibel eingesetzt werden kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

1.1 Ziel der Arbeit

Ziel dieser Arbeit ist es ein Framework zu konzipieren und erstellen. Dieses Framework soll für ein Communitynetzwerk erstellt werden. Im Zeitalter des Web 2.0, wo nahezu jeder einen Zugang zum schnellen Internet besitzt will sich jeder Mensch auch in diesem darstellen und mit anderen kommunizieren. Dies geschieht im Web 2.0 vor allem interaktiv in einer Online-Gemeinschaft. Das heißt, dass es nicht nur darum geht sich selbst, durch zum Beispiel Daten, Bilder oder Videos, darzustellen, sondern sich auch selber bei Anderen einzubringen. Dies kann in Form von Kommentaren zu Bildern und Videos sein, aber auch in Form von Nachrichtenaustausch und Bewertungen. Die Aufgabe des Frameworks ist es nun, ein Funktionsgrundgerüst zur Verfügung zu stellen, welches dem Anwender die Möglichkeit gibt, all dies zu tun also online interaktiv zu handeln. Das Framework muss dabei jedoch nicht nur flexibel gestaltet sein, sondern muss auch, da es sich um eine Unmenge von persönlichen Daten handelt, einem Anspruch auf Sicherheit gerecht werden.

1.2 Bedeutung von Netzwerken

Internetnetzwerke oder auch Onlinecommunitys haben durch die Weiterentwicklungen des Internets zum Web 2.0 einen enormen Schub erfahren. Seither gibt es immer Communities, die als Ziel haben, Menschen miteinander zu verbinden. Xing.net (Berufsnetzwerk), facebook.com (soziales Netzwerk) oder auch Myspace.com (soziales Netzwerk) sind dabei nur drei Beispiele von vielen. Seitdem die weltweite Vernetzung von Computern gestiegen und somit die Grundlage für soziale Netzwerke gegeben ist, treffen sich Menschen mit anderen in der virtuellen Welt, um zu kommunizieren. Über Profile und Bilder, die Benutzer selbst ins Netz stellen können, finden sich Menschen zu Interessengruppen zusammen und können so Meinungen und Erlebnisse mit-einander teilen. Seit Beginn der Onlinecommunitys veränderten sich die Plattformen dieser Gemeinschaften sehr stark. Bei den meisten Netzwerken haben die Mitglieder die Möglichkeit, mehr oder weniger Einfluss auf deren Entwicklung zu nehmen. Überwiegend ist es also kein zentrales Element, welches diesen Prozess vorantreibt, sondern dies geschieht vielmehr durch die Gemeinschaft selbst. So behauptet sich zunehmend ein Art „höhere Macht“ beziehungsweise Gerichtsbarkeit, die von den Mitgliedern der Community selbst geschaffenen und meist bei den Teilnehmern, Usern, erwünschte Gesetze durchsetzt.

Letztlich geht es darum, viele einzelne Benutzer zu einer großen Gemeinschaft zusammenzuschließen. Durch diese Kontakte und Verknüpfungen ist es möglich, Ziele und Wünsche jedes Mitgliedes zu erfüllen.

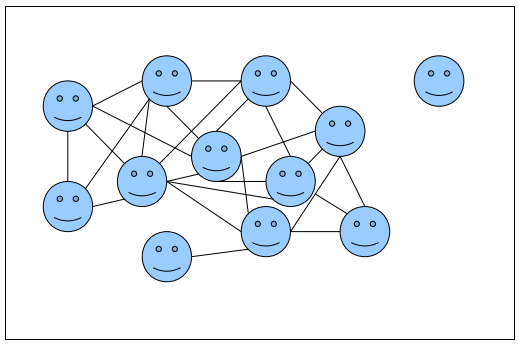


Abbildung 1: Soziales Netzwerk

1.3 Allgemeines zu Frameworks

Ein Framework ist ein Funktionsgrundgerüst. Dieses selbst stellt an sich keine Anwendung beziehungsweise kein System dar. Es wird dafür vielmehr eine Grundlage, eine „Rahmenstruktur“, geschaffen. Programmierer verwenden derartige Strukturen, um über den Gebrauch der bereitgestellten Funktionen Anwendungen zu erstellen. Das dem Framework zugrunde liegende Architekturmuster gibt die Anwendungsarchitektur vor. Diese „Gerüste“ gibt es in so ziemlich jeder Programmier- und Skriptsprache: zum Beispiel .NET- von Microsoft für C und C#, jQuery- von John Resig für Javascript- oder Zend Framework und CakePHP- für PHP. Im Gegensatz zu einer Klassenbibliothek, einer Sammlung einer Anzahl unterschiedlicher Klassen, macht ein Framework vor allem die Verbindung dieser Klassen, deren Methoden und Attribute sowie deren Schnittstellen deutlich und stellt damit einen klaren Vorteil heraus. Hauptaufgabe ist es also, bestimmte Funktionsabläufe in Klassen zu kapseln, um somit eine Programmierarchitektur zu schaffen, in der vor allem Muster von Abläufen wieder verwendet werden können. Diese bestimmten Vorlagen sind meist vom Anspruch oder vom Auftraggeber abhängig. Es gibt also bei jedem Framework eine bestimmte Beschränktheit in der Funktions- oder Arbeitsweise, was deren große Anzahl bestätigt (für PHP gibt es circa 20 Frameworks). Jedoch existieren auch einige grundlegende und vor allem von der Programmiersprache unabhängige Bedingungen, die ein Framework erfüllen muss, um möglichst optimal eingesetzt werden zu können:

- Einfachheit

  Ein Framework, vor allem das, welches es zu konzipieren und zu erstellen gilt, wird vor allem über seine Funktionsvielfalt bemessen. Es kann jedoch nicht das Ziel eines Entwicklers sein, eine große Anzahl an Funktionen zu implementieren und damit das Ziel und die Aufgaben des Frameworks aus den Augen zu lassen. In diesem Sinne ist es das Wichtigste, die konkreten Zielvorstellungen zu erfüllen und das Funktionsgrundgerüst in

diesem Fall so einfach und übersichtlich zu halten, wie es möglich ist.

- Erweiterbarkeit

  Um ein Framework in einem weiten Feld einsetzen zu können, ist es notwendig, bestimmte Funktionalitäten nachrüsten zu können. Bei der Erweiterbarkeit sollte es sich jedoch nicht nur um das rein additive Ergänzen handeln, sondern vor allem auch um das Modifizieren. So muss es Entwicklern möglich sein, das Framework so zu modulieren, dass am Ende die gewünschten Anwendungen erzielt werden können.

- Einsatz von Bewährtem

  Beim Designen eines Frameworks sollte nicht auf Techniken verzichtet werden, die sich seit längerem und mehrfach bewährt haben. Wenn es also möglich ist, derartige geeignete Software, Klassen, Funktionen oder Module einzubauen, ohne die Grundstruktur des Frameworks übermäßig zu beeinträchtigen, so sollte nicht darauf verzichtet werden. Der große Vorteil liegt darin, dass durch die Verwendung von probater Technik vor allem ein Zeitgewinn erzielt werden kann, da diese nicht einzeln sondern lediglich im Umgang mit den anderen Komponenten geplant, umgesetzt und getestet werden muss.

- Sicherheit

  Ein großes Thema bei der Konzeption und beim Design eines Frameworks sollte vor allem die Sicherheit darstellen. In vielen Fällen spielen sensible Daten eines Benutzers eine wichtige Rolle. Damit diese nicht in irgendeiner Form missbraucht, gelesen oder manipuliert werden können, sollte vor allem auf eine Validierung Wert gelegt werden. Lässt man diesen Punkt außer Acht, kann es schnell geschehen, dass der Anwendung und somit dem Framework kein Vertrauen geschenkt wird und dieses wiederum keine Verwendung mehr findet.

[Eder, S.1f]

2. Pflichtenheft

2.1 Zielbestimmung

Das Framework beziehungsweise das System bei dem das Framework integriert ist, wird verwendet für eine Internetplattform auf der sich Menschen treffen und über verschiedene Wege miteinander kommunizieren können. Ziel dieser Plattform ist es letztlich Menschen miteinander in Kontakt zu bringen und soziale Kontakte zu knüpfen.

2.1.1 Zielkriterien

Hierbei handelt es sich um Leistungen die für das Produkt von immenser Wichtigkeit sind. Diese Leistungen muss das fertige Produkt unbedingt enthalten beziehungsweise zur Verfügung stellen.

- Der Benutzer muss sich selbstständig am System registrieren können
- Der Benutzer soll sich selbstständig ein- und ausloggen können
- Der Benutzer soll die Möglichkeit haben, sein Profil anzulegen und auszufüllen
- Der Benutzer soll die Möglichkeit haben, die Profile anderer Mitglieder einzusehen
- Der Benutzer soll selbständig seinen Traumpartner und seine Vorlieben speichern können.
- Der Benutzer muss selbstständig Bilder hochladen können
- Der Benutzer soll mit anderen Kommunikation betreiben können
- Der Benutzer hat die Möglichkeit andere Mitglieder über eine Suchfunktion zu finden

2.1.2 Wunschkriterien

Bei den Wunschkriterien handelt es sich um Leistungen, die bei dem Fertigprodukt vorhanden sein können jedoch nicht müssen.
In diese Kategorie sind folgende Gesichtspunkte einzuordnen:

- Designer sollten die Möglichkeit haben ohne übermäßige Programmierkenntnise das Design zu erstellen beziehungsweise zu verändern.
- Programmierer sollten die Möglichkeit haben, das Framework ohne hohe Einarbeitungszeit zu erweitern oder zu verändern.
- Administratoren sollten die Möglichkeit haben, die Konfiguration des Systems an einer zentralen Stelle und somit ohne großen Aufwand zu verändern.
- Administratoren sollten die Möglichkeit haben, das System mit gerechtfertigtem Aufwand portieren zu können.

2.1.3 Use-Case-Darstellung

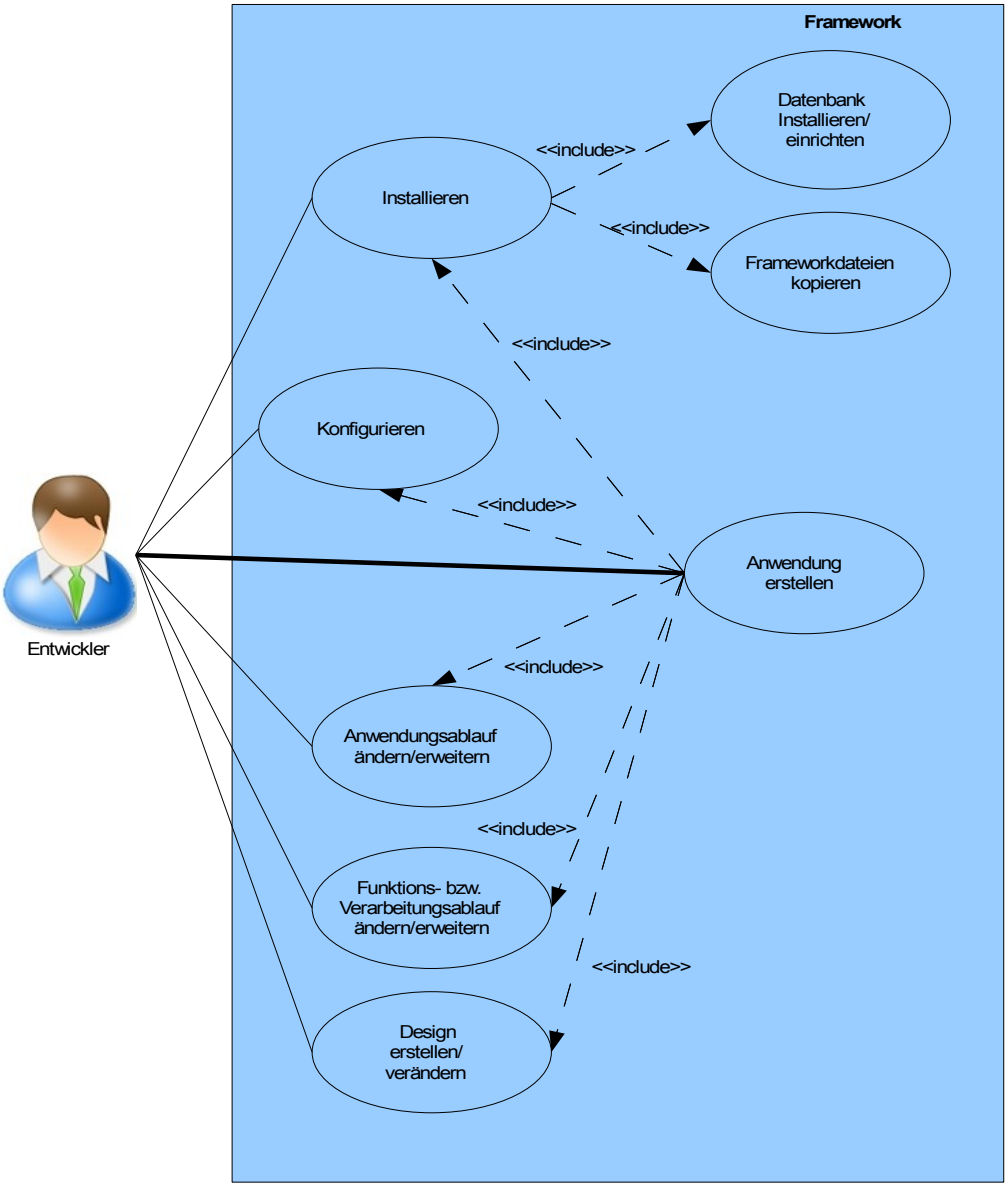


Abbildung 2: Use-Case-Darstellung

2.2 Produkteinsatz

2.2.1 Anwendungsbereiche

Das zu erstellende Framework soll für ein Community-Netzwerk eingesetzt werden.

2.2.2 Zielgruppen

Genutzt werden soll das Framework vor allem von Designern und Programmierern. Für Erstere darf dabei lediglich der Umgang mit HTML und Template Engines kein Problem darstellen.
Programmierer hingegen sollten ein mittleres Programmierniveau erreicht haben, um das Framework anzupassen oder gegebenenfalls erweitern zu können.

2.2.3 Betriebsbedingungen

Das System müsste täglich ununterbrochen 24 Stunden betriebsbereit und schon aus diesem Grund wartungsfrei und störunanfällig sein.
Der Administrator muss unter anderem Bilder, die hochgeladen werden, kontrollieren und freischalten können.
Es unterliegt ebenfalls dem Administrator, Benutzer zu sperren, die gegen die AGB des Systems verstoßen.

2.3 Produktumgebung

2.3.1 Software

Die Anforderung an die Produktumgebung für das System wird unterteilt in Client und Server:

Client – Softwareanforderung:

- Webbrowser der aktuellen Generation (z.B.: Internet Explorer 8, Firefox 3.x, Opera 9, Chrome)

Server – Softwareanforderung:

- Apache Webservers
- PHP (ab Version 5)
- MySQL-Datenbank (ab Version 5)
- SMTP – Server

2.3.2 Hardware

Auch im Bereich der Hardware wird zwischen Client und Server unterschieden.

Client – Hardwareanforderung:

- internetfähiger Computer

Server – Hardwareanforderung:

- netzwerk- bzw. internetfähiger Server
- gängige und aktuelle Webservertechniken (Arbeitsspeicher: ~4Gb,Festplatte:100Gb, CPU-Takt:ab 2Ghz)

2.4 Produktfunktionen

2.4.1 Registrierung

Die Registrierung für das System erfolgt im ersten Schritt mit der **E-Mail-Adresse**.
An diese Adresse wird dann eine Nachricht gesendet in der die Registrierung bestätigt wird.
Ist dies geschehen, muss der Nutzer noch die folgenden Daten eingeben um die Registrierung ab zu schließen:

- Passwort für die Anmeldung
- Benutzername
- Postleitzahl
- Geburtstag

Diese Daten können später gegebenenfalls in den Profileinstellungen noch geändert werden.

2.4.2 An- und Abmeldung

Die Anmeldung am System wird mit der registrierten E-Mail-Adresse und dem dazugehörigen Passwort vorgenommen. Dies setzt eine erfolgreiche Registrierung voraus. Nach der Anmeldung gelangt der Benutzer direkt in seinen Adminbereich. Von hier aus kommt er durch die Schnellnavigation zu jedem anderen Bereich; um zum Beispiel Nachrichten zu lesen oder zu schreiben oder Fotos hochzuladen.

2.4.3 Schnellnavigation

Die Schnellnavigation beziehungsweise das Menü ist ein Bereich, der für den Gebrauch stets angezeigt werden soll. Über das Menü soll es dem Benutzer möglich sein, zu jeder Zeit in ein anderes Modul zu gelangen und somit seine Daten oder die seiner Bekanntschaften ein zu sehen oder zu verändern.

2.4.4 Useradminbereich

Dieser Bereich dient dem Benutzer dazu seine Profildaten einzustellen beziehungsweise zu verändern. Von hier aus kommt er auch in den **Profilfoto**-Bereich um sein Profilbild zu erstellen oder zu verändern sowie in den Bereich für seine **Wishlist (Backstube)**. Im Useradminbereich kann der Benutzer seine Daten in folgenden Feldern speichern und verändern:

- Benutzername
- Statustext / Flirttext

- E-Mail-Adresse
- Geschlecht
- Name und Vorname
- Postleitzahl
- Geburtstag
- Größe
- Gewicht
- Haar- und Augenfarbe
- Raucherstatus
- Homepage
- Tätigkeit
- „Ich suche“

2.4.5 Profil

Im Profil kann der Benutzer seine Daten ansehen, wie er sie im Useradminbereich gespeichert hat.

Bereiche, die er nicht angegeben hat, werden nicht angezeigt. Als Besucher eines Profils kann man neben den Daten aus dem Useradminbereich auch das Profilbild sehen und zusätzlich hat man noch die Möglichkeit, auf die öffentlichen Bereiche des Benutzers zuzugreifen, wie zum Beispiel die Foto- und Videogalerie, die Freundesliste oder das Gästebuch.

Hier ist ebenfalls zu sehen, welche Nutzer zuletzt das Profil besuchten und sich dieses angeschaut haben. Als Benutzer kann man nicht nur das eigene Profil sehen, sondern vor allem auch die der anderen Mitglieder des Netzwerkes.

2.4.6 Voting

Weiterhin findet man im Bereich Profil die Benutzerbewertung. Hierbei ist es anderen Nutzern möglich, ein Profil beziehungsweise den Benutzer selbst zu bewerten. Dabei wird geschlechtsspezifisch unterschieden, das heißt danach, wie ein männlicher Benutzer von anderen männlichen beziehungsweise von anderen weiblichen Benutzern bewertet wurde.

2.4.7 Profilfoto

Das Profilfoto dient zur optischen Erkennung des Benutzers. Über den Useradminbereich ist es möglich ein Profilbild in drei Schritten hochzuladen.

1. Bild hochladen
2. Bildausschnitt auswählen
3. Profilbild speichern

Wird für ein vorhandenes Profilbild ein neues hochgeladen, wird das Profilbild ersetzt und steht dementsprechend nicht mehr als Profilbild zur Verfügung.

2.4.8 Wishlist/ Backstube

Die Wishlist oder auch Backstube ist ein Fragekatalog den sich der Benutzer beantworten kann um

1. seine Angaben zu verfeinern und
2. um die Angaben bezüglich seines Traumpartners an zu geben.

Die Fragen sind dabei die gleichen und dienen letztlich dazu, über Vergleiche der Angaben passende Partner herauszufinden.

2.4.9 Fotogalerie

Im Bereich der Fotogalerie ist dem Benutzer folgendes möglich:

- Fotoalben anlegen
- Fotoalbum löschen
- pro Fotoalbum eine unbegrenzte Anzahl an Bildern hochladen
- Bilder eines Albums in einer Vorschau anzeigen lassen (Thumbnail-Galerie)
- Bilder in voller Größe anzeigen lassen
- jedes Bild mit eigenem Namen versehen
- andere Benutzer (Freunde) auf den Bildern verlinken
- Kommentare zu den Bildern hinzufügen
- Kommentar(e) löschen

Einige Funktionen stehen dabei jeweils nur dem Besitzer der Fotogalerie zur Verfügung. Dazu gehören das Anlegen und Löschen von Fotoalben, das Hochladen von Bildern sowie das Ändern der Bildnamen. Um einen Kommentar zu löschen, muss man entweder der Besitzer der Galerie oder aber der Autor des Kommentars sein.

2.4.10 Videogalerie

Die Videogalerie funktioniert analog der Fotogalerie. Der Benutzer kann, allerdings nur wenn er der Besitzer der Galerie ist, neue Videos zu seiner Galerie hinzufügen. Dieses funktioniert über ein Eingabefeld und kann Videolinks der Plattform YouTube verarbeiten. Weiterhin ist es für alle Benutzer, Besucher und Besitzer, möglich, einen Kommentar zu einem Video abzugeben. Wie bei der Foto- kann auch bei der Videogalerie nur deren Besitzer Daten entfernen beziehungsweise ausschließlich der Autor des Kommentars diesen löschen. Doch im Gegensatz zur Fotogalerie kann der Benutzer hier keine Alben anlegen.

2.4.11 FensterIn

Entweder ein männlicher Benutzer klopft an das Fenster eines weiblichen und sendet eine FensterIn-Anfrage- also ob sie ihm das Fenster öffnet und er ihr Nachrichten schreiben darf- oder sie wählt ihn aus, als ob sie ihm das Fenster geöffnet hat. Dann erhält er eine Statusnachricht.

Der Benutzer kann die FensterIn-Anfragen in seinem Adminbereich verwalten. Wenn einem männlichen User der Zugang gestattet ist, dann kann er beim Besuchen des Profils der Userin das geöffnete Profil sehen und kann somit Kontakt mit der Benutzerin aufnehmen.

2.4.12 Freundesliste

In dem System ist es möglich, andere Benutzer als Freunde zur eigenen Freundesliste hinzuzufügen.

Hat man als Nutzer über die Suche jemanden gefunden, kann man an diesen einen Antrag stellen mit dem Status Bekanntschaft, Freundschaft, Date, Bekannter, Verwandter oder Ähnliches. Der Empfänger kann dann entscheiden, ob er diesen annimmt, ablehnt oder einen Gegenantrag stellt. Wird der Antrag angenommen, kann der Benutzer über die Freundesliste dann direkt auf das Profil des Freundes zugreifen.

2.4.13 Nachrichtenaustausch (Nachrichtengruppe)

Für die Kommunikation zwischen den Benutzern dient das Nachrichtensystem. Jeder Nutzer hat seinen eigenen Nachrichtenbereich, den nur er einsehen kann. In diesem Bereich ist es dem Benutzer möglich, Nachrichten zu lesen und zu schreiben sowie auf bereits versendete Nachrichten zuzugreifen (Nachrichtenausgang). Im Nachrichtenein- und -ausgang ist es dem Benutzer möglich, einzelne Meldungen zu löschen.

Weiterhin besteht für ihn die Möglichkeit, Nachrichtengruppen anzulegen und zu verwalten. Diese Gruppen dienen dazu, sogenannte „Rundmails“ zu versenden, das heißt der Benutzer muss beim Verfassen der Nachricht nicht alle Adressaten einzeln angeben, sondern kann sie vorher sammeln und muss dann lediglich die Gruppe angeben.
Ein weiterer Bereich ist die sogenannte „Sozialakte“. In dieser können Anträge auf Entsperrung verwaltet und bearbeitet werden.

2.4.14 VIP-Bereich

Dieser Bereich gibt dem Benutzer die Möglichkeit, eine Sammlung von Bildern, Texten, Videos, Terminen usw. anzulegen und diese mit einem Passwort zu versehen, welches der Benutzer selber erstellen kann. So kann man als Besucher nur auf diese Daten zugreifen, wenn man das Passwort kennt.
Die Möglichkeit, einen VIP-Bereich einzurichten, existiert nur für Mitglieder, die eine hohe Aktivität haben.

2.4.15 Gästebuch

Als Besucher eines Profils kann man, ähnlich den Kommentaren bei der Video- und Fotogalerie, einen Eintrag im Gästebuch hinterlassen. Wie dabei üblich, sind diese Einträge öffentlich, das heißt sie können von jedem anderen Besucher gelesen werden. Der Besitzer des Gästebuches kann einzelne Einträge löschen oder kommentieren. Wenn ein Eintrag „gesprengt“, das heißt vom Besitzer gelöscht wird, dann bleibt noch ersichtlich, wer den Eintrag geschrieben hat, jedoch nicht was er geschrieben hat. Vom System erfolgt dann automatisch eine Systemnachricht an den Autor des Eintrages.

2.4.16 Suche (Schnellsuche, Übereinstimmungssuche, Zufallssuche)

Bei der Suche gibt es drei Möglichkeiten:

1. Suche nach einem Profil mit Hilfe des Benutzernames

   Hierbei wird lediglich der Benutzername eingegeben und das System sucht dann nach Mitgliedern, die diesen oder einen ähnlichen Namen haben.

2. gezielte Suche mit Hilfe von Angabe einiger Eigenschaften

   Mit dieser Suchmöglichkeit kann der Benutzer andere Mitglieder finden, indem er zum Beispiel angibt, welches Alter diese ungefähr haben sollen oder aus welcher Stadt sie kommen. Dabei sind auch Kombinationen von Eigenschaften möglich

(Bsp: über 20 Jahre, Dresden, weiblich → alle Frauen die über 20 Jahre alt sind und aus Dresden kommen.)

3. Zufallssuche (Blinddate)

Die Zufallssuche bietet für den Benutzer die Möglichkeit, ein „Blinddate“ finden zu lassen. Es werden dabei die Wishlist- beziehungsweise Backstubendaten von anderen Mitgliedern mit denen des Benutzers verglichen. Aus diesem Vergleich wird dann die Zusammengehörigkeit ermittelt, das heißt wie gut die beiden Mitglieder zusammenpassen.

Aus den Mitgliedern mit hoher Übereinstimmungsrate wird eine Liste gebildet und per Zufall einer ausgesucht, mit dem dann (nur) Nachrichten ausgetauscht werden können. Als weiteres Auswahlkriterium dient die Bedingung, dass nie zweimal das gleiche Blinddate durchgeführt wird.

Während der Kommunikation zwischen beiden, können sie sich die Benutzernamen austauschen, da keiner der beiden die Möglichkeit hat das Profil des anderen einzusehen. Der Austausch des Benutzernamens ist jedoch freiwillig.

Die Ergebnisse der gezielten und einfachen Suche werden mit Benutzername, Foto, Ort und Alter angegeben und dabei wird auch ausgelesen, was der Nutzer in seinem Suchprofil angegeben hat. Sie können nach Übereinstimmung, Alter, Anzahl der Freunde sortiert werden.

2.4.17 Steckbrief/Profil pausieren

Diese Funktion gibt dem Mitglied die Möglichkeit, die Mitgliedschaft ohne Löschung seiner Daten zu beenden. Das hat einen entscheidenden Vorteil: Sollte er wieder Mitglied werden wollen, muss er nicht erneut sämtliche Daten eingeben. Hat der Benutzer den o.g. Modus aktiviert, wird das in keiner Freundesliste beziehungsweise in keinem Suchergebnis angezeigt. Außerdem ist es keinem anderen Mitglied möglich, auf das Profil des pausierenden Nutzers zuzugreifen. Er selbst hat jedoch jederzeit die Möglichkeit, sein Profil zu reaktivieren und er wird dann auch wieder als normales Mitglied geführt. Der Rückweg, die erneute Aktivierung und das Verlassen des „Pause“-Modus, führt über den Menüpunkt „Einstellungen“. Sämtliche anderen Funktionen stehen dem Benutzer, der pausiert, nicht zur Verfügung.

2.4.18 (Eulen und Uhu schießen)

Das Spiel „Eulenschießen“ funktioniert folgendermaßen:
Als Spieler hat man drei verschiedene Wurfobjekte: Herz, Blume, faules Obst. Nach dem Zufallsprinzip erscheinen dann Profilbilder von anderen Benutzern, die mit diesen Objekten beworfen werden können. Nach Genauigkeit der Treffer erhält der Spieler Punkte. Je mehr der Benutzer in dem Spiel sammelt, desto höher ist seine Aktivität. Ist diese ausreichend hoch, kann er proportional dazu Zusatzfunktionen des Systems nutzen, zum Beispiel den VIP-Bereich. Dieses Spiel soll als Flashanwendung erstellt werden und ist nicht Teil der Erstellung und Konzeption.

2.4.19 Systemnachrichten

Um den Benutzer über Neuigkeiten zu informieren, erhält er über das System und über die eingegebene E-Mail-Adresse Nachrichten mit den nötigen Informationen, beispielsweise „BenutzerXY hat ein neues Fotoalbum“ oder „Du hast neue Nachrichten“.

Dabei handelt es sich um Neuigkeiten, die ihn betreffen, wie zum Beispiel

- Nachrichten,
- Gästebucheinträge,
- Freundschaftsanträge,
- Freischaltanträge,
- Verlinkungen auf Bilder,
- Kommentare zu Bildern oder Videos

oder wo es um seine Freunde geht:

- neue Bilder oder Bilderalben
- Änderung des Kontaktstatus , z.B.von „Date“ auf „Freundschaft“

Diese Nachrichten erhält der Benutzer dann per E-Mail oder aber auf seiner Startseite mit dem Link zu dem betroffenen Bereich.
Er kann jedoch, wenn er keine E-Mail oder Systemnachricht erhalten will, diese jeweils deaktivieren.

2.4.20 Einstellungen(Sperrverwaltung, Stalkerverwaltung, Systemnachrichten)

Über den Menüpunkt „Einstellungen“ kann der Benutzer die Regulierungen für die Sperrverwaltung, die Stalkerverwaltung und für die Systemnachrichten vornehmen.

1. Sperrverwaltung:

   Die Sperrverwaltung hat die Funktion, die folgenden öffentlich zugänglichen Bereiche bei Bedarf zu sperren:

   - Onlineliste
   - Profile
   - Gästebuch
   - Kommentare
   - Verlinkungen auf Fotos
   - Videogalerie und
   - Fotogalerie

   Die Sperrung kann entweder offen oder geheim sein.

   Offen bedeutet, dass Besucher dieses Bereichs die Mitteilung erhalten, dass jener nicht angezeigt wird, weil dessen Besitzer ihn gesperrt hat. Bei der geheimen Sperrung hingegen erhält der Besucher keinerlei Erklärung. Falls ein Bereich für die Öffentlichkeit gesperrt wurde, ist es dem Besucher möglich, einen Antrag auf „Entsperrung“ oder „Freischaltung“ zu stellen. In diesem muss der Antragsteller seinen Wunsch bzw. seine Bitte begründen. Dieser Vorgang erscheint dann im Bereich Nachrichten im Ordner „Sozialakte“ und kann vom Benutzer bearbeitet werden. Sollte der Nutzer den Besucher freischalten, also entsperren, erhält Letzterer eine Systemnachricht

2. Stalkerverwaltung

   Mit der Stalkerverwaltung wird die Absicht verfolgt, dass ein Benutzer einen anderen ignorieren kann. Diesmal wird unterschieden in offenes oder diskretes „Nichtbeachten“.

   Ist die Stalkerverwaltung im Modus offen geschaltet, kann die ignorierte Person keine Nachrichten mehr an den Benutzer senden. Außerdem ist es für das ignorierte Mitglied weder möglich einen Gästebucheintrag vorzunehmen bzw. Kommentare zu Videos oder Bildern zu verfassen noch das Profil oder den Onlinestatus des Benutzers einzusehen.

   Bei der diskreten Stalkerverwaltung werden lediglich die Nachrichten der ignorierten Person automatisch in den Papierkorb verschoben. Es können aber weiterhin

Kommentare geschrieben und das Profil eingesehen werden.

3. Sonstiges Einstellungen

Im Bereich der sonstigen Einstellungen kann der Benutzer regeln, ob er eine eintreffende Systemnachricht als E-Mail, als Nachricht auf seiner Startseite oder gar nicht erhalten will.

Auf dieser Ebene findet sich auch der Regulierungsmechanismus für das Profil „Pausieren“ (siehe oben).

2.5 Produktdaten

Folgende Daten werden durch das Framework verarbeitet beziehungsweise gespeichert:

2.5.1 Systemdaten:

Diese Daten werden speziell für das eigene System gespeichert und werden auch nur von diesem verwendet.

Mitglieder:

E-Mail-Adresse
Passwort
Geburtstag
Postleitzahl
Aktivierungs-Bit (Der Benutzer ist aktiviert oder nicht)
Registrierungsdatum

Wishlist/Backstube – Auswahlfelder:

Name des Auswahlfeldes
deutscher Titel des Auswahlfeldes
englischer Titel des Auswahlfeldes

Wishlist/Backstube – Optionen:

Name des Auswahlfeldes zu dem die Option gehört
Wert der Option
Anzeigename der Option (deutsch und englisch)

2.5.2 Benutzerdaten:

Die nun folgenden Daten beziehen sich speziell auf den Benutzer. Alle Daten die hier gespeichert werden, können vom Benutzer in den bestimmten Feldern in der Benutzeransicht verändert und verarbeitet werden.

Freundesliste:

- Freund (wer ist der Freund)
- Freundschaftsstatus (Beziehung, Bekanntschaft)

Gästebuch:

- Text des Eintrags
- Zeit des Eintrags
- Autor des Eintrags
- Kommentar des Gästebuchbesitzers

Ignorierverwaltung:

- Id des Benutzers (Wer ignoriert?)
- ignorierter Benutzers (Wer wird ignoriert?)
- Status der Ignorierung

Bilder:

- Id des Eigentümers des Bildes
- Albenname
- Bildpfad
- Bildname (standardmäßig: Name des hochgeladenen Bildes)

Bilderkommentare:

- Bild-Id (Zu welchem Bild wurde ein Kommentar abgegeben?)
- Kommentartext zu dem Bild
- Kommentarzeit
- Kommentarautor (Id)

Benutzerverlinkung:

- Bild-Id
- Verlinkter Benutzer (Id)
- Benutzer der Verlinkt hat (Id)
- Position der Verlinkung auf dem Bild in Pixel (horizontale und vertikale Position)
- Verlinkungszeit
- Freischaltungs-Bit

Sperrverwaltung (allgemeine Einstellung):

Sperrtyp (offen oder geheim)

Benutzer-Id

Freischaltungs-Bit für

Onlinestatus, Gästebuch, Profil, Video- und Bildergalerie, Bildverlinkung

Sperrverwaltung (Benutzerspezifische Freischaltung):

Freischalt-Begründungstext

Benutzer-Id

Id des freigeschalteten Benutzers

Freischaltungs-Bit für

Onlinestatus, Gästebuch, Profil, Video- und Bildergalerie, Bildverlinkung

Zeitpunkt der letzten Änderung

Nachrichtengruppen:

Besitzer der Gruppe (Id des Benutzers)

Gruppenname

Gruppenmitglieds-Id

Nachrichten:

Absender-Id

Empfänger-Id

Nachrichtentitel

Nachrichtentext

Archivierungsbit

Lösch-Bit des Absenders

Lösch-Bit des Empfängers

Datum der letzten Änderung

„Gelesen“-Status-Bit

Blinddate-Bit

Freischalt-Antrags-Bit

Profildaten:

Benutzer-Id

Name

Vorname

Benutzername

Geschlecht
Stadt
Größe
Gewicht
Augenfarbe
Homepage
Arbeit /Ausbildungsstatus
Raucherstatus
Datum der letzten Änderung
Onlinestatus
Sperr-Bit für E-Mail- bzw. Systemnachrichten
Profilbildpfad
Statustext
Besucher
Pausestatus
Wishliste/ „Backstube“-Daten des Benutzers

Profilbewertung:

Benutzer-Id
Bewerter-Id
Bewertergeschlecht
Wert der Bewertung

Systemnachrichten:

Benutzer-Id
Systemnachrichtentext
Zeitpunkt der Systemnachrichten
„Gelesen“-Status-Bit
Modulname
Link zu Modul

Video:

Benutzer-Id
YouTube-Video-Id
Videotitel
Zeitpunkt der Speicherung

Videokommentare:

Kommentarautor (Id)

Videobesitzer (Id)

Kommentartext

Kommentarzeit

2.6 Produktleistungen

Die Produktleistungen, sind die Leistungen, die das System beziehungsweise im engeren Sinne das Framework automatisch erbringt. Diese Funktionen arbeiten vom Nutzer unabhängig, das heißt, dass keine Eingaben oder Tätigkeiten vom Nutzer benötigt werden.

2.6.1 Methode der „sprechende URL“

Das System unterstützt die Methode der „sprechenden URL“. Das heißt, dass der Benutzer anhand der URL schon genau erkennen wo er sich befindet. Ganz besonders für Suchmaschinen sind die URLs geeignet, da diese nun nicht mehr die Parameter sehen die mit übergeben werden sondern eine einfach und klare URL.

Beispiel: www.beispiel.de?user=Name wird dementsprechend zu www.beispiel.de/Name

Speziell für das System ist die Vorgaben so, dass die URL in einer bestimmten Reihenfolge aufgebaut sein soll:

1. Domain und Top Level Domain
2. /Name des Benutzers
3. /Modul welches gerade verwendet wird
4. /weitere Unterelemente, je nachdem welches Modul verwendet wird

Beispiel für den Aufbau der URL:

www.single.by/user1/fotos/MeinFotoalbum/Urlaubsbild1

Eine weitere Anforderung an die URL ist, dass sie nicht nur deutsche Module, sondern gegebenenfalls auch englische Modulnamen anzeigen soll, also beispielsweise statt „*nachrichten“* - „*messages“*.

2.6.2 Traumpartner/Traumpaar

Das System kann aufgrund der Benutzerangaben, das heißt derjenigen über sich selbst und über seine Vorstellungen den Traumpartner betreffend, passende Kombinationen von Mitgliedern finden. Diese Daten werden im Modul Wishlist /“Backstube“ vom Benutzer angegeben. Einen Vergleich aus den Angaben der „Backstube“ des einen Benutzers mit denen aller anderen durchzuführen, ist eine weitere Aufgabe des Systems. Deren Ergebnis wird dann im Anschluss nach Übereinstimmungen geordnet. Somit kann der Benutzer einsehen, wer zu ihm besser passt und wer nicht.

2.6.3 Keine doppelte Bewertung

Bei der Bewertung eines Mitglieds auf dessen Profil überwacht das System diesen Vorgang so, dass dieses durch keinen Benutzer mehrfach bewertet werden kann. Somit wird eine Verfälschung der Gesamtbeurteilung ausgeschlossen.

2.6.4 Bilderfreigabe

Da der Inhalt von Bildern, die von Benutzern hochgeladen werden, nicht vom System geprüft werden kann, wird es eine Möglichkeit für den Administrator geben, Bilder frei zu schalten. Das heißt, der Administrator oder die Administratorengruppe kann dann per Bildereinsicht die Bilder freischalten.

2.7 Benutzungsoberfläche

2.7.1 Benutzerführung

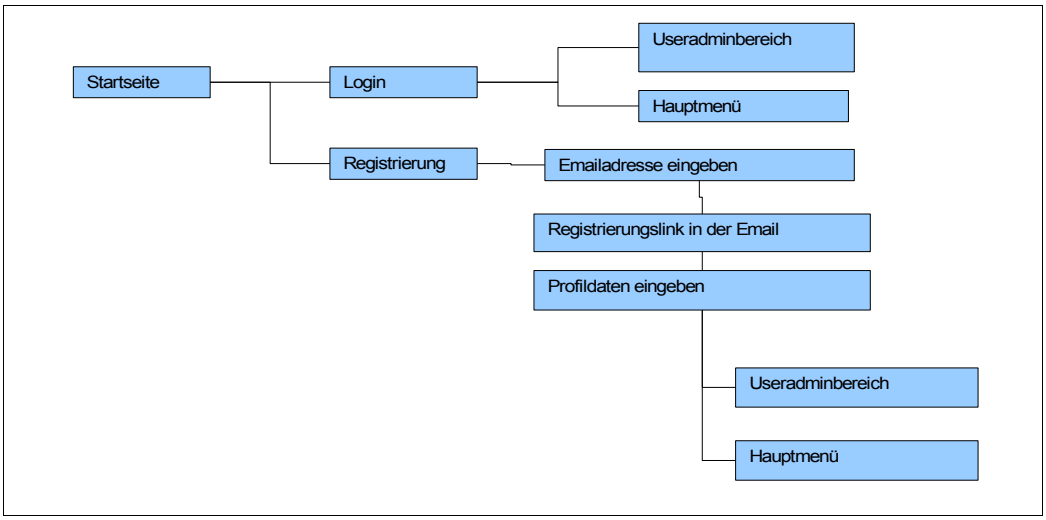


Abbildung 3: Benutzerführung

2.7.2 Hauptmenü

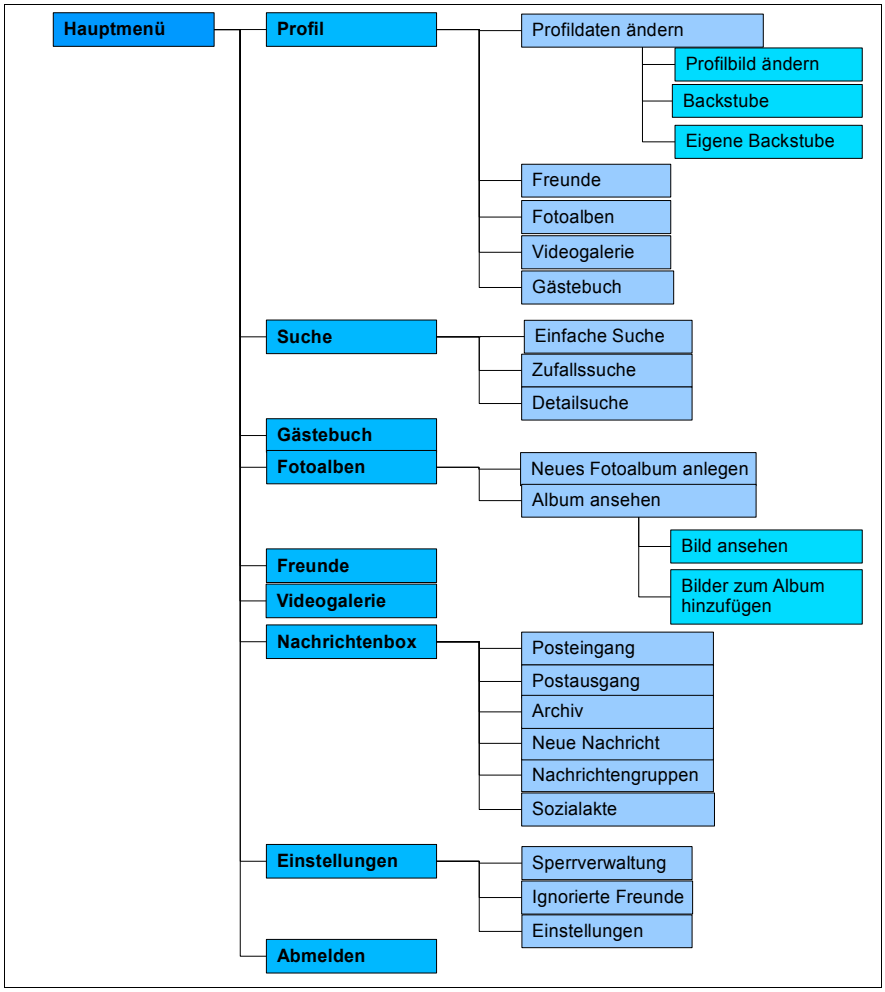


Abbildung 4: Hauptmenüstruktur

2.8 Entwicklungsumgebung

Für die Entwicklung des Frameworks braucht man neben einem Apache-Webserver mit PHP 5 auch eine MySQL-Datenbank. Eine Entwicklungsumgebung, z.B. ein Notepad, und vorzugsweise eine PHP-unterstützende IDE, z.B. Notepad++ oder PHPEclipse, wird weiterhin für die reine Entwicklung benötigt.
Es wird empfohlen einen der gängigen aktuellen Browser (Internet Explorer 8, Firefox 3.x, Opera) zu verwenden. Weiterhin wird für das Datenbankmanagement die Software phpMyadmin, sowie HeidiSQL (eine quelloffene Software für MySQL) verwendet.

2.9 Ergänzungen

Das System kann mit mehreren Sprachen betrieben werden. Dazu zählt neben der deutschen - im Spezialfall bayerisch - auch die englische. Für die Umstellung der Sprache wird jeweils ein Modul vorhanden sein, welches man bei Bedarf gegen ein anderes austauschen kann. Die einzelnen Sprachmodule können jeweils erweitert bzw. erneuert werden.
Wichtig für die Verwendung des Frameworks ist die Portier- und Erweiterbarkeit. Das Framework soll für mehrere Communities genutzt werden und in diesen für den Benutzer unterschiedliche Module anbieten.
Für einige Bereiche ist es nötig, dass der Benutzer Java Script zur optimalen Nutzung aktiviert hat. Dies gilt vor allem für die Bilder- beziehungsweise Fotogaleriebereiche.

3. Entwurf und Konzeption

Das Framework soll bestimmte Aufgaben und Vorgaben erfüllen und natürlich nicht nur zur Erstellung einer, sondern mehrerer Anwendungen dienen. Es muss also flexibel einsetz- und erweiterbar sein. Um die Erfüllung solcher unterschiedlicher Anforderung bei jeder einzelnen Verwendung zu gewährleisten, ist die Entwicklung eines derartigen Frameworks erforderlich, welchem ein komponentenorientierter Entwicklungsansatz zugrunde liegt.

3.1 Das MVC-Entwurfsmuster

Um das Framework ohne übermäßig zusätzlichen Zeitaufwand für Designer und Programmierer in verschiedenen Systemen einzusetzen, ist es erforderlich, die Programmlogik so strikt wie möglich vom Design beziehungsweise der Benutzerschnittstelle zu trennen. Es empfiehlt sich hier die Verwendung des MVC-Entwurfsmusters, um dies konsequent durchzusetzen.
Das bezeichnete Muster ist ein Konzept für die Softwareentwicklung. Es teilt die Anwendung in drei mehr oder weniger separate Einheiten ein: in Model, View und Controller. Für die Planung und Entwicklung größerer Softwareprojekte bietet das MVC-Konzept viele Vorteile:

- *Trennung von Logik und Design*

  Die Trennung von Logik und Design, gibt den Programmierern und Designern die Möglichkeit, unabhängig voneinander zu arbeiten. Somit brauchen sie sich jeweils nur mit einem Gebiet, Logik oder Design, konkret auseinandersetzen und es findet keine Vermischung dieser beiden Bereiche statt. Der Designer muss sich also lediglich mit der grafischen Aufarbeitung beschäftigen und der Logiker beziehungsweise Programmierer mit der Verarbeitung der Informationen.

- *Möglichkeit der parallelen Entwicklung von Elementen*

  Die Anwendung oder das Projekt wird durch das MVC-Prinzip in drei unabhängige Elemente geteilt. Dadurch ist es möglich, dass jedes dieser Elemente, also Model, View und Controller, getrennt von den anderen erstellt oder bearbeitet werden kann.

- *Komponenten können besser erweitert beziehungsweise ausgetauscht werden*

  Das MVC-Entwurfsmuster bietet weiterhin den riesigen Vorteil, dass Elemente nicht nur getrennt voneinander, sondern auch wesentlich günstiger erweitert oder sogar ganz ausgetauscht werden können. In dem zu erstellenden Framework bringt gerade dieser Punkt einen großen Gewinn. Denn so kann bei dessen Verwendung die Funktionalität in verschiedenen Anwendungen eingesetzt werden. Für diese muss man dann jeweils nur noch die grafische Benutzerschnittstelle (GUI) anpassen.

- *Design beinhaltet nur noch sehr wenig Skript-Code*

  Für die Designer hat das Entwurfsmuster einen zusätzlichen Vorzug: Sie haben sich nur noch mit dem Code für den Browser, für HTML, CSS oder JavaScript, auseinanderzusetzen. Dabei muss lediglich auf den Code von Template Engines, wie Smarty oder PHP Template Engine, geachtet werden. Diese dienen allerdings auch dazu, die Informationen anzuzeigen, welche als Antwort vom Server gesendet werden.

  Ein Beispiel für ein HTML-Template mit PHP-Template Engine Code

```
<html>
   <head>
      <title><?=$title?></title>
   </head>
   <body>
      <p><?=$text?></p>
   </body>
</html>
```

Die folgende Grafik verdeutlicht den grundlegenden Aufbau des MVC-Prinzips, wobei diese grafische Darstellung für eine Webanwendung optimiert ist. Das MVC-Prinzip jedoch funktioniert

auch bei jeglichen anderen Softwareprojekten.

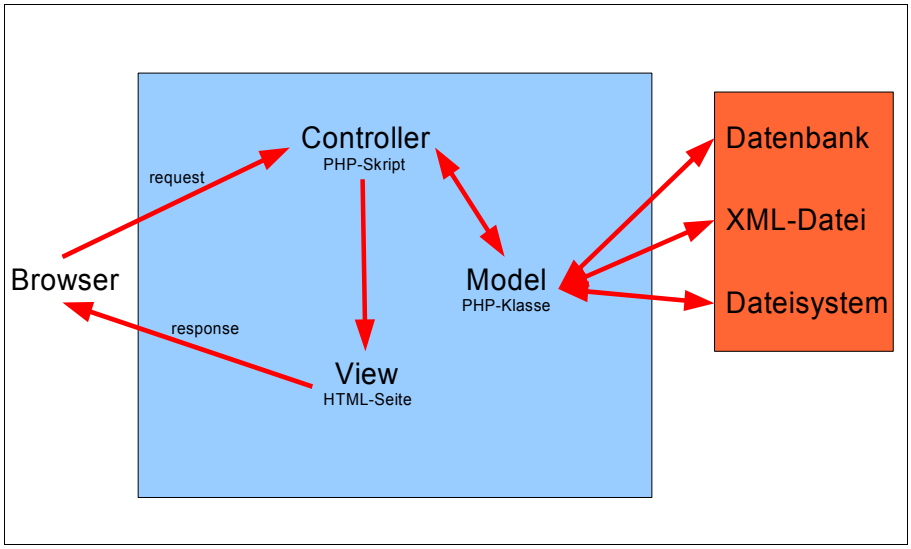


Abbildung 5: MVC-Konzept

Die Hauptkomponenten in dieser Übersicht haben jeweils unterschiedliche Aufgaben und Funktionen:

Model

Das Model enthält die eigentliche Programmlogik und die Daten, welche dargestellt werden sollen. Diese Schicht umfasst Methoden, die Daten zu manipulieren. Eine weitere ihrer Aufgabe ist es, die Informationen der Anwendung zum Beispiel in Datenbanken oder im Serverdateisystem zu speichern und daraus auch wieder auszulesen. Die Modelschicht arbeitet unabhängig von den beiden anderen Schichten.

View

Die Viewschicht dient der Präsentation der Daten des Programms beziehungsweise der Anwendung. Hierbei handelt es sich also um die reine Darstellung von Informationen und nicht um die Verarbeitung von Daten in irgendeiner Art und Weise. Neben diesem Darstellen werden die Views vor allem auch dafür gebraucht, Interaktionen vom Benutzer entgegenzunehmen und die dadurch entstehenden Daten wiederum an einen Controller weiterzuleiten. Im speziellen Fall einer Webanwendung bestehen die Views vor allem aus HTML-, CSS-, Javascript- und Template Engine-Codes.

Controller

Die Controllerschicht ist das Element im MVC-Konzept, welches die Steuerung zwischen View- und Modelschicht übernimmt. Diese Schicht stellt also die zentrale Verwaltung dar, welche nicht nur die Benutzerdaten der View empfängt, sondern aufgrund dieser Daten auch darüber entscheidet, welches Model diese weiterverarbeitet. Die Ergebnisse der

Datenmanipulation helfen wiederum dem Controller zu entscheiden, welche View aufgerufen wird.

Je nachdem wie bedeutend die Benutzerangaben für die Modelschicht sind, wird entweder im Controller oder erst im Model eine Datenvalidierung durchgeführt. Daten zu verarbeiten beziehungsweise zu manipulieren, ist jedoch definitiv keine Funktion des Controllers.

Die grafische Übersicht enthält noch zwei weitere Elemente, die hier nur kurz angesprochen werden sollen:

Ein Element davon ist der **Browser**, welcher unter anderem dafür zuständig ist, Benutzerangaben an den Server zu senden, "request" genannt. Er stellt dabei eine grafische Benutzerschnittstelle dar, über die der Nutzer Eingaben vornehmen kann. Eine andere Aufgabe des Browsers ist es, die vom Server gesendeten Informationen, "response", grafisch aufzuarbeiten und für den Benutzer darzustellen.

Das zweite Element aus der Übersicht sind die **externen Dateioperationen**. Damit ist der Austausch von Informationen von einem Model-Element mit externen Dateien oder Datenbanken gemeint. Neben dem Lesen und Schreiben in eine XML-Datei oder in eine MySQL-Datenbank zum Beispiel ist es auch möglich, auf das Dateisystem des Servers zuzugreifen. Dies geschieht unter anderem dann, wenn beispielsweise Bilder auf den Server geladen, von diesem wieder gelesen oder gelöscht werden.

3.2 Erweitertes MVC-Konzept

Auf Grund der Größe des Projektes und der damit verbundenen Anzahl der Module ist es für das Framework sehr von Vorteil, das bereits beschriebene MVC-Modell zu erweitern.

Im Falle des zu planenden Frameworks wird die Controller-Ebene um einen Zusatz erweitert. Die beiden anderen Ebenen, Model und View, bleiben allerdings unverändert.

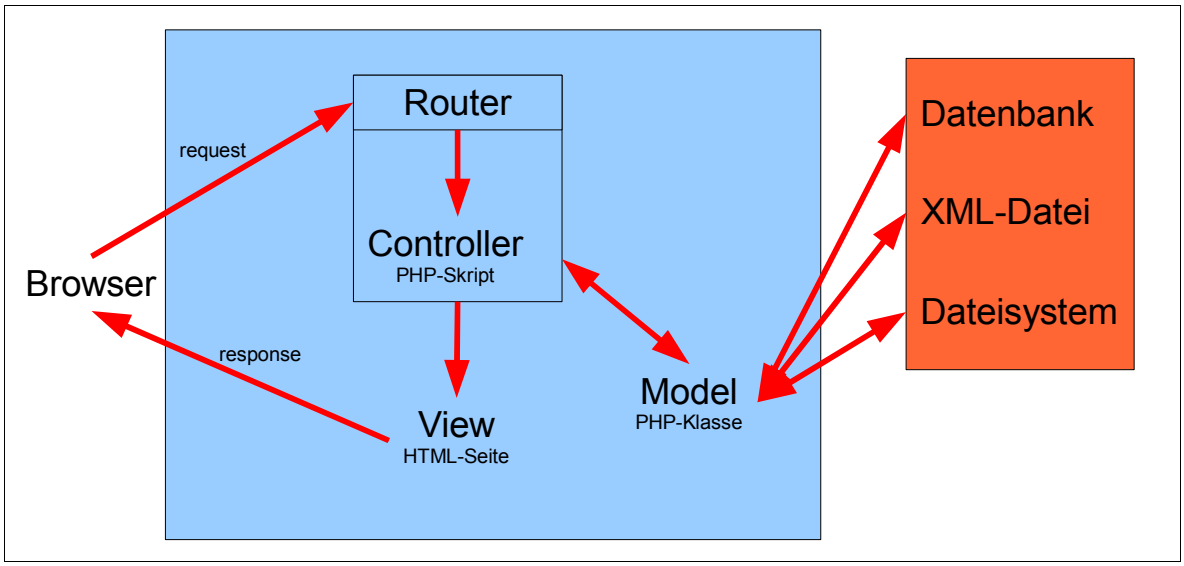


Abbildung 6: Erweitertes MVC-Konzept

Wie in der Grafik zu sehen ist, wird vor dem Controller ein sogenannter Router, ein Verteiler, eingesetzt. Dieser hat die Aufgabe, je nach Anfrage des Browsers, "request", einen Controller auszuwählen, der diese entgegen nimmt und weiterverarbeitet. Der Vorteil dieser Vorgehensweise liegt darin, dass neben der bereits bestehenden Möglichkeit der parallelen Arbeit an der Model- und View-Ebene nun auch gleichzeitig an mehreren Controllern gearbeitet werden kann. Hier gilt natürlich ebenfalls, dass durch die Teilung der Controller-Ebene nun auch dort sowohl paralleles Arbeiten möglich ist als auch besserer Austausch beziehungsweise Bearbeitung einzelner Controller nach Fertigstellung des Frameworks .

Die Weiterverarbeitung des Requests, nachdem dieser durch den Router geleitet wurde, erfolgt wie bereits weiter oben erwähnt.

3.3 Modulare Bauweise

Das Framework muss so gestaltet sein, dass dieses eine weitverbreitete Verwendung finden kann. Dazu ist nicht nur dessen Flexibilität vonnöten, sondern auch eine beliebige Erweiterbarkeit. Das bedeutet, dass das Framework aus einzelnen Teilbereichen, sogenannten Modulen, bestehen muss um so eine klare Trennung zwischen den einzelnen Bereichen zu sichern.

3.4 Klassenübersicht

Um die Klassenbibliothek für das Framework zu planen, ist es zuerst wichtig, die Anforderungen an dieses zu analysieren. Danach gilt es, die Funktionalitäten den Anforderungen entsprechend so aufzuteilen beziehungsweise zu kapseln, dass letztlich Erweiterungen ohne größeren Aufwand möglich sind und dennoch keine Einschränkungen (der Funktionalität) vorgenommen werden müssen.Die Verwendung von Klassen ist an dieser Stelle angebracht, da jenes Softwarekonstrukt die Möglichkeit bietet, Funktionalitäten beziehungsweise Methoden und Eigenschaften zu kapseln.

Folgende Punkte stellen ebenfalls noch wichtige Vorzüge für den Einsatz der objektorientierten Programmierung dar:

- *Ordnung*

  Die Verwendung von Klassen eröffnet die Möglichkeit, Funktionalitäten zu sammeln und zu ordnen. Daraus ergibt sich für Entwickler und Programmierer, die das Framework verwenden, die Chance, sich besser in dieses selbst und dessen Funktionsweise einzuarbeiten.

- *Erweiterbarkeit und Flexibilität*

  Auf Grund der Kapselung von Methoden und Eigenschaften in Klassen und der dadurch entstehenden Ordnung ist es Programmierern möglich, Funktionalitäten einfacher zu erweitern und so die Flexibilität zu steigern.

- *Wiederverwendbarkeit*

  Die Wiederverwendbarkeit von Methoden beziehungsweise Verarbeitungsabläufen ist durch den Gebrauch der objektorientierten Programmierung leichter gegeben, als dies bei einer einfachen, losen Sammlung von Funktionen der Fall wäre.

Um die Anforderungen und Aufgaben des Frameworks zu ver- beziehungsweise bearbeiten, ist es notwendig, folgende Klassen in die Klassenbibliothek aufzunehmen:

- Datenbankklasse

  Die Datenbankklasse enthält Methoden, die das Framework mit der Datenbank kommunizieren lassen. Dazu gehören neben derjenigen, welche die Verbindung zur Datenbank herstellt, noch die vier grundlegenden Datenbankoperationen:

  - lesen (*read*)
  - aktualisieren (*update*)
  - löschen (*delete*)
  - einfügen (*insert*)

  Diese dienen dazu, Informationen mit der Datenbank auszutauschen - im Sinne von lesen, löschen, einfügen usw. Darüber hinaus muss diese Klasse eine weitere Methode bereithalten, die den Datenbankrückgabewert verarbeiten und weiterleiten kann. Das heißt, wenn Daten aus der Datenbank gelesen werden, nimmt sie das Ergebnis - *resource* beziehungsweise *resultset* im Falle einer "lesen"-Anweisung - auf und bearbeitet es so weiter, dass letztlich ein Array oder String zurückgegeben wird. Die Datenbankklasse ist die wichtigste der Klassenbibliothek, da durch sie die Informationen und Zustände dauerhaft in

einer Datenbank gespeichert werden können.

- Datei- und Dateisystemklasse

  Um auf das Serverdateisystem und damit auf Dateien zugreifen zu können, werden entsprechende Methoden benötigt. Diese stellt die Datei- beziehungsweise Dateisystemklasse zur Verfügung. Jene wird vorrangig für das Organisieren von Bilddateien in Ordner verwendet, d.h. zum Einfügen, Lesen und Löschen. Um nun auf alle Dateioperationen vorbereitet zu sein, sollten in der Dateisystemklasse derartige Methoden bereitstehen, die die folgenden Aufgaben bearbeiten können:

  - Verzeichnis erstellen

  Im Bereich der Bildergalerie hat der Benutzer die Möglichkeit, Alben anzulegen. Diese werden im Dateisystem des Servers als Verzeichnis dargestellt. Um also diesen Auftrag bewältigen zu können, muss die Dateisystemklasse eine Methode des Erstellens von Verzeichnissen bereithalten.

  - Verzeichnis auslesen

  Um zum Beispiel auf alle Bilder eines Verzeichnisses zugreifen zu können, muss eben dieses ausgelesen und die Informationen, wie Namen und Typen der Dateien, müssen gesammelt werden. Der Rückgabetyp ist ein Array, weil man damit leichter auf die gesammelten Informationen zugreifen kann.

  - Unterverzeichnisse auflisten

  Die Vorgehensweise ist bei dieser Aufgabe ähnlich der vorangegangenen. Es müssen in einem Verzeichnis alle Unterverzeichnisse aufgelistet werden, also deren Namen. Die Liste wird dann als Array zurückgegeben.

  - Verzeichnis löschen

  Neben dem Erstellen und Auslesen von Verzeichnissen muss es in der Dateisystemklasse auch eine Methode geben, diese zu löschen. Jener Vorgang wird zwei Teile beinhalten. Zunächst müssen alle Inhalte des zu löschenden Verzeichnisses entfernt werden. Erst danach kann der Ordner vom Server entfernt werden.

  - Datei(en) löschen

  Um ein Verzeichnis oder auch nur jeweils eine Datei beziehungsweise ein Bild zu entfernen, muss die Klasse eine Methode zum Löschen einzelner Dateien bereithalten.

- Freundeklasse

  Die Freundeklasse verarbeitet die Beziehungen eines Benutzers allen anderen gegenüber.

Das heißt, hier werden alle Aktionen vollzogen, die den Benutzer passiv gesehen betreffen. Diese sind im Folgenden aufgelistet:

- Auslesen aller Freunde / Beziehungen

  Diese Methode soll alle Beziehungen eines Benutzers mit Angabe des Beziehungsstatus sowie mit einigen Detailangaben des Freundes auslesen.

- Ignorierstatus setzen und lesen

  Hierbei handelt es sich um Methoden, die den Status einer Ignorierung des Benutzers setzten beziehungsweise aus der Datenbank auslesen können. Das heißt, die Methoden können den Status setzten und auch wieder löschen. Beim lesen des Status geht es darum, ob beziehungsweise was genau ein Besucher auf einer Seite eines Benutzers sehen darf. Weiterhin wird es eine Methode geben, die, für die Verwaltung, alle ignorierten Bekanntschaften eines Benutzers ausliest und diese Informationen dann zur Auflistung zur Verfügung stellt.

- Sperrstatus setzen und lesen / Sperrverwaltung

  Wie bei der Ignorierverwaltung so geht es auch bei der Sperrverwaltung darum, gesperrte Bereiche zu verwalten. Das heißt, dass bestimmte Bereiche vom Benutzer gesperrt oder entsperrt werden können. Zudem werden Methoden bereit gehalten, die eventuelle Freischaltungen verarbeiten können.

- Bilderklasse

  Die Aufgaben der Bilderklasse ist es Methoden zur Verfügung zu stellen, die jedes Vorhaben im Bezug auf Bilder behandelt. Folgende Aufgaben sind dabei zu verarbeiten:

  - Prüfung des Dateiformates

    Um Bilder ordnungsgemäß zu verarbeiten, muss rein technisch gesehen die Sicherheit gegeben werden. Dies wird vorerst du eine Überprüfung der Dateiendung vorgenommen. Geprüft werden die Formate *jpeg* (Joint Photographic Experts Group), *png* (Portable Network Graphics) und *gif* (Graphics Interchange Format).

  - Profilbildupload

    Für das Hochladen eines Profilbildes ist eine spezielle Methode vorgesehen, da damit die Größe des Bildes sowie der Ort des Hochladens direkt angegeben werden kann.

  - allgemeine Bilduploadmethode

    Dies soll die Methode werden, welche den allgemeinen Bilderupload übernimmt. Sie findet vor allem Verwendung im Bereich der Bildergalerie. Die Hauptaufgaben der

Methode, sind

- Bilder temporär auf den Server hoch laden
- Bildgröße ändern
- Bildpfad anpassen und bereitstellen
- Thumbnail erstellen

◦ Bilderdaten laden

Die Aufgabe dieser Methode soll es sein, die Daten eines Bildes zu laden. Das heißt, wenn ein Bild angezeigt werden soll, gibt diese Methode folgende Daten aus:

- Bildname
- Kommentare des Bildes mit Kommentarautor, Kommentarzeit und Kommentartext
- Bildverlinkungen(Position, Verlinkter Benutzer)

◦ Bilderliste bereitstellen

Bei der Anzeige eines Albums kommt unter anderen diese Methode zum Einsatz. Sie stellt, alle Bilder eines Albums zu Verfügung. Dabei werden die Informationen über Bildname, Pfad sowie Anzahl aller Bilder ausgegeben. Letzteres wird für die Auflistung aller Bilder benötigt (für die Darstellung in einer geordneten Übersicht).

- E-Mailklasse

Die Versendung von E-Mails in jeglicher Form wird von der E-Mailklasse übernommen. Das Zusammensetzen von Texten und versenden von bestimmten E-Mailformen ist die Hauptaufgabe dieser Klasse.

- Nachrichtenklasse

Die Nachrichtenklasse übernimmt das Versenden und Anzeigen von Nachrichten des Benutzers an andere Mitglieder des Netzwerkes. Folgende Grundaufgaben müssen dabei bearbeitet werden:

◦ Nachricht versenden, also Text und Absender sowie Adressat in Datenbank eintragen

◦ Empfänger der Nachricht als Benutzer-Id bereit stellen

◦ Nachrichten mit Absender, Zeit und Text laden (Postaus- und Posteingang, Archiv)

◦ Nachrichten löschen

◦ Antwortdaten laden

- "Sozialakte" laden

- Nachrichtengruppenklasse

  Für die Erstellung und Verwaltung von Nachrichtengruppen wird die Nachrichtengruppenklasse verwendet.

  Folgende Aufgaben werden dabei verarbeitet:

  - Nachrichtengruppe erstellen
  - Nachrichtengruppe löschen
  - Alle Nachrichtengruppen eines Benutzers auflisten
  - Mitglieder einer Nachrichtengruppe laden
  - Benutzer aus einer Nachrichtengruppe löschen

- Registrierungsklasse

  Für alles was die Registrierung betrifft wird die Registrierungsklasse in Anspruch genommen. Es gilt dabei, folgende Vorgaben zu erfüllen:

  - Eintrag in Mitgliederdatenbank
  - Benutzerkonto anlegen
  - Bestätigungsmail absenden
  - temporäres Passwort generieren

- Systemnachrichtenklasse

  Die Systemnachrichten dienen dem Benutzer zur Bereitstellung von Informationen die direkt den Benutzer vom System her betreffen. Dabei muss auf folgende Anforderungen Rücksicht genommen werden:

  - Neue Nachricht anlegen (in Datenbank eintragen und E-Mail versenden)
  - Status der Systemnachricht ändern (auf "gelesen" zum Beispiel)
  - Alle Systemnachrichten des Benutzers auslesen

- Benutzer- und Benutzerdatenklasse

  Diese Klasse ist neben der Datenbankklasse mit eine der wichtigsten Klassen. Sie verarbeitet jegliche Benutzeraktionen und Benutzerdaten. So gehören folgende Aufgaben zur dieser Klasse:

- Login-Vorgang
- Logout-Vorgang
- Profildaten auslesen, ändern, speichern
- Besucherliste laden und speichern
- Gästebuchdaten verwalten (auslesen, löschen, speichern)
- Freischalten einer Verlinkung auf einem Bild

- Validierungsklasse

  Die Validierungsklasse übernimmt sämtliche Aufgaben zur Validierung von Formularen und Ähnlichem. Neben der Überprüfung auf Vollständigkeit werden auch folgende Bestandteile auf Validität überprüft:

  - E-Mailadressen (@-Zeichen, Domain, Umlaute)
  - Geburtsdatum (Datumsformat, nur Zahlen)
  - Passwort (Länge, Beschaffenheit aus Zahlen und Buchstaben)
  - Benutzername (keine vorbelegten oder gegen AGB verstoßende Benutzernamen)
  - Postleitzahl (5-stellig, keine Buchstaben
  - Login (Abgleich von Logindaten und Passwort mit Datenbank)
  - Nachrichtenüberprüfung (Anzeige der nicht ausgefüllten Felder)

  Eine weitere Aufgabe dieser Klasse ist es, Einträge vom Benutzer in das UTF-8 Format um zu wandeln und darüber hinaus HTML-Tags sowie unnötige Leerzeichen zu entfernen.

- XML-Klasse

  Die XML-Klasse findet vor allem bei der Verarbeitung und dem Auslesen von XML-Dateien Verwendung. Neben dem Bereitstellen von Informationen aus einer XML-Datei über DOM oder XPath ist es eine Aufgabe auch Daten in eine XML-Datei zu schreiben. Unterstütz werden die genannten Vorgänge von SimpleXML, einem Modul welches bereits in PHP5 enthalten ist.

- Votingklasse

  Laut Vorgabe sollte es eine Funktionalität geben als Benutzer einen anderem Benutzer zu bewerten. Dies wird mit der Votingklasse sicher gestellt. Die Hauptaufgaben diese Klasse sind folgende:

- Auslesen der Bewertungen (getter-Methode)
- Speichern von Bewertungen (setter-Methode)

Hierbei sind jedoch einige wichtige Punkte zu beachten. Neben der normalen Speicherung des Bewertung muss auch gespeichert werden wer und somit welches Geschlecht die Bewertung vorgenommen hat. Ersteres dient dabei lediglich der Absicherung, dass niemand das Bewertungssystem durch Dauerbewertung verfälscht. Das Geschlecht dient hingegen nur der Auswertung an sich (Wie bewerten weibliche Benutzer einen männlichen und umgekehrt).

- Videoklasse

  Für den Bereich der Videogalerie wird die Videoklasse eingesetzt. Die Aufgabe dieser ist es, die Videos die ein Benutzer gespeichert hat zu verwalten. Dazu gehört

  - Speichern eines Videos mit Hilfe der Youtubeklasse
  - Zusammenstellen einer Liste mit allen gespeicherten Videos
  - Auslesen aller Informationen über das aktuelle Video (Kommentare, Titel, Video-Id)
  - Aufbereiten dieser Informationen

- Youtubeklasse

  Laut aktuellen Vorgaben sollen nur Videos gespeichert beziehungsweise verarbeitet werden, die von der Onlinevideoplattform YouTube stammen. Um nun die Möglichkeit offen zu lassen, noch andere Onlinevideoplattformen zu zulassen und damit eine gewisse Modularität zu schaffen ist es besser die Verarbeitung von Informationen über die Videos in Videoplattformabhängigen Klassen unter zu bringen. Deswegen gibt es neben der Videoklasse noch eine weitere Klasse, die ebenfalls zu dem Bereich der Videogalerie gehört. Aufgabe dieses Softwarekonstruktes ist es nun die Informationen über ein Video von youtube.com auszulesen und weiter zu verarbeiten. Die Aufgaben dieser Klasse gliedern sich konkret wie folgt:

  - Youtubevideo-Id aus der URL auslesen
  - XML-Datei von youtube.com über die ID laden
  - konkrete Informationen aus der XML-Datei auslesen (Videothumbnail, Titel)

  Sollten nun noch anderen Videoplattformen zugelassen werden, können diese mit den gleichen Aufträgen versehen werden.

3.5 Die MySQL-Datenbank

Die Datenbank spielt für ziemlich jedes System die zentrale Rolle, wenn es darum geht, Zustände und Informationen dauerhaft und in geordneter Form zu speichern. Neben der Speicherung jedoch, bietet sie noch die Möglichkeit, die gespeicherten Informationen zu verwalten, d.h. zu löschen oder zu verändern, und die Daten in ihrer geordneten Form zu sichern.

Diese Möglichkeiten der Datenspeicherung und -verwaltung bietet neben anderen Datenbanksystemen, wie PostgreSQL, Oracle und MS SQL, auch eine MySQL-Datenbank an. MySQL bietet den großen Vorteil, dass es im Open-Source-Sektor am weitesten verbreitet ist und somit auch eine große Nutzergemeinde hat. Dies wiederum bringt den Gewinn, dass bei der Arbeit mit MySQL auftretende Probleme wesentlich schneller gelöst werden können. MySQL hat allerdings noch einige weitere positive Eigenschaften. Es ist

- schnell,
- stabil,
- plattformunabhängig (läuft auf Windows, Linux, Mac OS X und Unix-Derivaten),
- umfassend dokumentiert,
- unter GPL (General Public License) frei verfügbar,
- Programmiersprachenkompatibel (PHP, Java, C, C++, C#, Perl, Python, VB).

Für den Programmierer bietet MySQL eine große Anzahl an Funktionen zum Verwalten und Bearbeiten der gespeicherten Daten. [Kofler, S.23]

Im Folgenden wird die Planung der einzelnen Tabellen in der Datenbank aufgezeigt. Hierbei sind für jede Tabelle eine kleine Übersicht gegeben und besondere Attribute kenntlich gemacht worden.

3.5.1 Mitglieder

Name	**Typ**	**Attribute**
ID	Zahl (integer)	Primary Key, auto_increment
email	Ziffern und Buchstaben (varchar)	unique
password	Ziffern und Buchstaben (varchar)	
birthday	Datum (date)	
localID	Zahl (integer / varchar)	
acitvated	Zahl (integer)	
registrationDate	Datum und Zeit (datetime)	

3.5.2 Profil

Name	Typ	Attribute
userID	Zahl (integer)	Primary Key
gender	Text und Zahlen (varchar)	
name	Text und Zahlen (varchar)	
forename	Text und Zahlen (varchar)	
city	Text und Zahlen (varchar)	
height	Text und Zahlen (varchar)	
weight	Text und Zahlen (varchar)	
hairClr	Text und Zahlen (varchar)	
eyeClr	Text und Zahlen (varchar)	
smoke	Text und Zahlen (varchar)	
homepage	Text und Zahlen (varchar)	
work	Text und Zahlen (varchar)	
lookFor	Text und Zahlen (varchar)	
userName	Text und Zahlen (varchar)	unique
lastAction	Zeitstempel (timestamp)	
online	Zahl (integer)	
emailmessages	Zahl (integer)	
systemmessages	Zahl (integer)	
img	Text und Zahlen (varchar)	
friends	Text und Zahlen (varchar)	
statusText	Text und Zahlen (varchar)	
visitors	Text und Zahlen (varchar)	
wishlist	Text und Zahlen (varchar)	
wishlist_self	Text und Zahlen (varchar)	
pause	Zahl (integer)	

3.5.3 Nachrichten

Name	Typ	Attribute
messageId	Zahl (integer)	Primary Key, auto_increment
sender	Text und Zahlen (varchar)	
acceptor	Text und Zahlen (varchar)	
title	Text und Zahlen (varchar)	
message	Text und Zahlen (varchar)	
archived	Zahl (integer)	
deleted_acceptor	Zahl (integer)	
deleted_sender	Zahl (integer)	
lastAction	Zeitstempel (timestamp)	
groupMessageId	Zahl (integer)	
readStatus	Zahl (integer)	
blinddate	Zahl (integer)	
requestMessage	Zahl (integer)	

3.5.4 Nachrichtengruppen

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	*unique*
member	Zahl (integer)	
groupName	Text und Zahlen (varchar)	

3.5.5 Freunde

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	unique
friend	Zahl (integer)	
relation	Text und Zahlen (varchar)	
confirmed	Zahl (integer)	

3.5.6 Gästebuch

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
owner	Zahl (integer)	
writer	Zahl (integer)	
writeTime	Zahl (integer)	
entry	Text und Zahlen (varchar)	
comment	Text und Zahlen (varchar)	

3.5.7 Bilder

Name	Typ	Attribute
imageId	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	
album	Text und Zahlen (varchar)	
image	Text und Zahlen (varchar)	
imgName	Text und Zahlen (varchar)	
unlocked	Text und Zahlen (varchar)	

3.5.8 Bilderkommentare

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
imgId	Zahl (integer)	
userId	Zahl (integer)	
comment	Text und Zahlen (varchar)	
commentTime	Text und Zahlen (varchar)	

3.5.9 Verlinkungen

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	unique
imgId	Zahl (integer)	
posx	Zahl (integer)	
posy	Zahl (integer)	
linkFrom	Zahl (integer)	
linkTime	Zahl (integer)	
unlocked	Zahl (integer)	

3.5.10 Video

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	
videoId	Text und Zahlen (varchar)	
videoTitle	Text und Zahlen (varchar)	
uploadedTime	Zahl (integer)	

3.5.11 Videokommentare

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
videoId	Zahl (integer)	
userId	Zahl (integer)	
comment	Text und Zahlen (varchar)	
commentTime	Text und Zahlen (varchar)	

3.5.12 Voting/Bewertung

Name	Typ	Attribute
userId	Zahl (integer)	unique
voterId	Zahl (integer)	
voterSex	Text und Zahlen (varchar)	
voteValue	Text und Zahlen (varchar)	

3.5.13 Ignorierung

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	
ignored	Zahl (integer)	
typ	Text und Zahlen (varchar)	

3.5.14 Sperrung

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	unique
onlinestatus	Zahl (integer)	
guestbook	Zahl (integer)	
profil	Zahl (integer)	
profilview	Zahl (integer)	
galery	Zahl (integer)	
imagelinks	Zahl (integer)	
wishlist	Text und Zahlen (varchar)	
videos	Zahl (integer)	
locktype	Text und Zahlen (varchar)	

3.5.15 Entsperrte Benutzer

Name	Typ	Attribute
user	Zahl (integer)	unique
unlocked_userid	Zahl (integer)	
requestMessage	Text und Zahlen (varchar)	
onlinestatus	Zahl (integer)	
guestbook	Zahl (integer)	
profil	Zahl (integer)	
imagelinks	Zahl (integer)	
galery	Zahl (integer)	
videos	Zahl (integer)	
confirmed	Zahl (integer)	
lastAction	Zeitstempel (timestamp)	

3.5.16 Systemnachrichten

Name	Typ	Attribute
id	Zahl (integer)	Primary Key, auto_increment
user	Zahl (integer)	
message	Text und Zahlen (varchar)	
messageTime	Zahl (integer)	
readStatus	Zahl (integer)	
modul	Text und Zahlen (varchar)	
link	Text und Zahlen (varchar)	

3.6 Die Konfiguration

Um das Framework möglichst flexibel einsetzen zu können und eine hohe Erweiterbarkeit zu erreichen, muss es entsprechend konfigurierbar sein. Ein weiterer wichtiger Punkt in diesem Sinne ist, dass die Konfiguration an einem zentralen Punkt vorliegen muss, damit Programmierer oder Administratoren leicht darauf zugreifen können.

Die Konfiguration beinhaltet grundsätzlich alle Parameter, die für den Einsatz des Frameworks auf einer Plattform nötig sind. Je nachdem also, welche Plattform beziehungsweise welche Grundvoraussetzungen oder auch Anforderungen gegeben sind, d.h. welche Datenbank, welches Dateisystem, welche Sprache und so weiter vorliegen, muss lediglich die Konfiguration angepasst werden.

Diese wurde in zwei separate Teile aufgespaltet, um die genannten Anforderungen erfolgreich zu bewältigen. Auf der einen Seite werden vom Framework Parameter benötigt, damit dieses auf der Plattform funktioniert. Die andere Seite jedoch betrifft solche, die weniger dem Funktionieren des Systems dienen als vielmehr dazu, dem Benutzer zum Beispiel eine andere Sprache anzuzeigen. Es muss also eine Systemkonfiguration und eine solche für die Oberfläche beziehungsweise das Design geben.

3.6.1 Die Systemkonfiguration

Die Systemkonfiguration beinhaltet Parameter, die für den Betrieb des Systems, in dem das Framework verwendet wird von sehr großer Bedeutung sind.

Dazu gehören:

- Datenbankverbindung,
- Datenbanktabellenamen,
- Pfadangaben von Ordnern,
- Klassenpfade,
- eventuell weitere Pfade,
- Bildergrößen (Höhe, Breite).

Da diese Parameter zur Laufzeit, das heißt während des Betriebs des Systems, vorhanden und dem Framework bekannt sein müssen, ist es von Vorteil, diese bereits zum Start des Systems als Konstanten zur Verfügung zu haben.

3.6.2 Die Oberflächenkonfiguration

Die Oberflächenkonfiguration enthält im Gegensatz zu der des Systems keine Parameter, die für Letzteres wichtig sind, sondern vor allem solche, die für den Nutzer Bedeutung haben. Dabei kann man noch einmal zwischen den Parametern unterscheiden, die zur Laufzeit dem Framework bereits bekannt sein müssen, und denen, die erst je nach Benutzeraktion zur Verfügung stehen sollen. Zu den zuerst erwähnten gehören zum Beispiel die verschiedensprachigen Modulnamen. Diese werden, ähnlich der Systemkonfiguration, als Konstante definiert und sind so bereits zum Start der Anwendung dem Framework bekannt.

Die anderen, von der Benutzeraktion abhängigen Parameter sind zum Beispiel:

- Auswahl- und Optionenfelder,
- Systemnachrichten,
- Textbausteine.

Diese Parameter werden in einer XML-Datei deklariert. Der Nutzen dieser Verfahrensweise ergibt sich daraus, dass sie dann in XML-Form zur Verfügung stehen. Dies bietet wiederum folgende Vorteile:

- Die Parameter stehen in einer geordneten und übersichtlichen Form zur Verfügung, was vor allem dazu dient Parameter hinzuzufügen, zu erweitern oder zu verändern.
- Über das XML-Format kann man direkt auf die benötigten Daten zugreifen. So müssen nicht immer alle Parameter gleichzeitig geladen werden, sondern nur die, die das Framework gerade braucht.

4. Implementierung

4.1 Dateisystem

Bei der Erstellung eines Frameworks und vor allem bei dessen Aufbau ist es wichtig, bestimmte Bereiche, Design und Funktion zum Beispiel, programmiertechnisch voneinander zu trennen. So erreicht man dessen größtmögliche Flexibilität und Erweiterbarkeit. Um diesen Anforderungen beziehungsweise Vorgaben über die gesamte Entwicklung hinweg konstant gerecht zu werden, muss man diesen Ansatz der Trennung von bestimmten Bereichen auch im Dateisystem, dem Positionsort der Klassendateien, Skripte sowie Templates, fortführen. Das MVC-Konzept teilt das Framework in die drei Bereich Model, View und Controller. Für Administratoren und Programmierer ist es also nützlich, jene auch im Dateisystem wiederzufinden. Zusätzlich zu diesen Räumen muss es jedoch noch andere geben, in denen zum Beispiel Konfigurationsdateien lagern. Schließlich ist es sinnvoll, die Klassendateien und Skripte in folgendem System abzulegen:

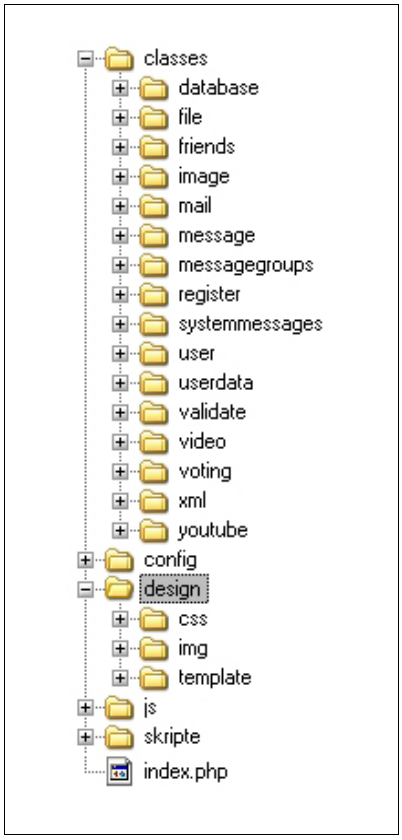


Abbildung 7: Dateisystem

In der Übersicht sind folgende Bereiche enthalten:

- classes — In diesem Ordner liegen - in ihren jeweiligen Unterordnern - die Klassen (=Model).
- config — Dies ist der zentrale Ort, wo die Konfigurationsdateien liegen.

- design Der Ordner repräsentiert den View-Bereich. Unterteilt wird hier noch einmal in Unterordner für die Stylesheetdefinitionen, für Bilder und für die Templates.
- js In diesem Ordner befinden sich alle Javascriptdateien.
- skripte Der Controller-Bereich in dem alle Skripte verfügbar sein werden.

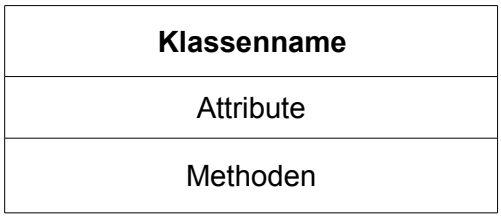
4.2 Programmiersprache

Für die Umsetzung der Anforderungen stehen verschiedene Programmiersprachen zur Verfügung. Darunter fallen vor allem die bekannten Sprachen Python, PHP, .NET und Java. Jede dieser Sprachen hat seine speziellen Vor- und Nachteile. Letztlich spielte jedoch nur die Entscheidung des Auftraggebers eine Rolle und dieser entschied sich für PHP als Sprache für die Umsetzung. Entscheidende Faktoren sind dabei vor allem das Know-How des Auftraggebers bezüglich dieser Sprache sowie der Aufwand und damit die Kosten der Bereitstellung der Laufzeitumgebung für PHP. Ein weiterer Grund ist, dass PHP mit zu den Sprachen gehört, die nur von sehr wenigen Komponenten abhängig sind, sei es nun ein zu Grunde liegendes Frameworks oder das Entwicklerwerkzeug (im Gegensatz zu .NET oder Java). Die große PHP-Nutzergemeinde und die damit zusammenhängende Unterstützung für Projekte ist ein weiterer großer Vorteil von PHP.

4.3 Klassen

4.3.1 UML-Klassendiagramm

Dieser Abschnitt beschäftigt sich mit der Darstellung der Klassenbibliothek als UML-Diagramm (Unified Modeling Language). Ein Klassendiagramm nach UML hat grundlegend den folgenden Aufbau:



Die Attribute werden untereinander als eine Kette von Zeichen dargestellt. Als erstes erfolgt dabei die Darstellung der Sichtbarkeit des Attributes: "-" für `private` und "+" für `public`. Danach folgt der Name sowie der Datentyp des Attributes. Ein Attribut kann dementsprechend so angegeben werden:

```
+ number : int
```

Die Angabe der Methoden einer Klasse ist ähnlich. Es wird lediglich die Sichtbarkeit und der Name der Methode angegeben, wobei die gleichen Vorgaben gelten, wie bei den Attributen[Spolwig]:

- countNumbers()

Es folgen nun die Darstellungen der Klassen aus der Klassenbibliothek des Frameworks als UML-Klassendiagramm:

Die Datenbankklasse:

Database
- conn : resource - static instance : object
+ __construct() + static getInstance() - connect() + insert() + read() + update() + delete() + delAlbum() + delImage() + getMessageStatus() + getUserId() + getUserName() - returnHandler()

Abbildung 8: Datenbankklasse

Die Dateisystemklasse:

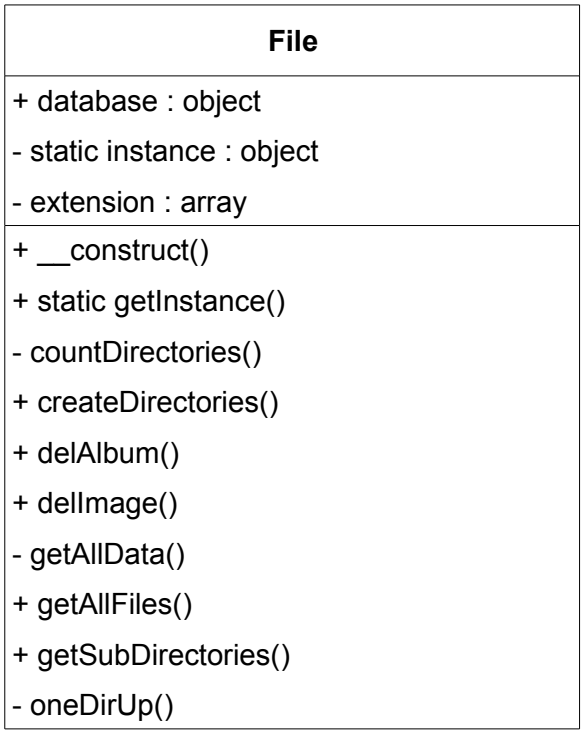


Abbildung 9: Dateissystemklasse

Die Freundeklasse:

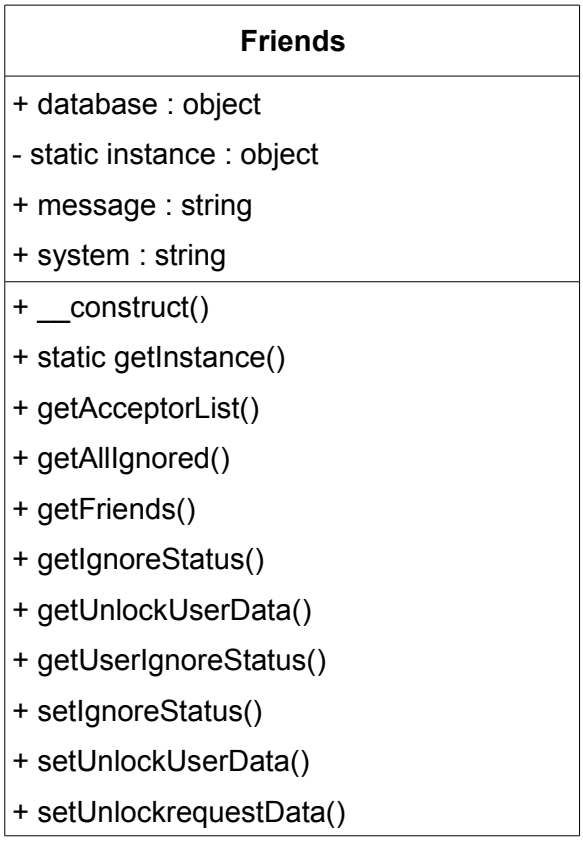


Abbildung 10: Freundeklasse

Die Bilderklasse:

Abbildung 11: Bilderklasse

Die E-Mailklasse:

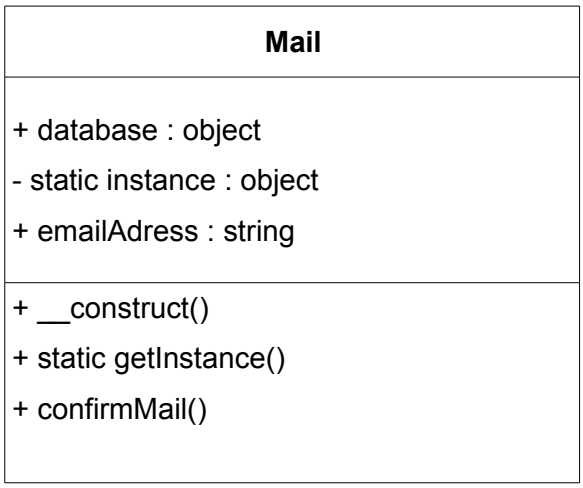


Abbildung 12: E-Mailklasse

Die Nachrichtenklasse:

Abbildung 13: Nachrichtenklasse

Die Nachrichtengruppenklasse:

Messagegroups
- database : object - static instance : object - friendsClass : object - system : object
+ __construct() + static getInstance() + createMessagegroup() + deleteMessageGroup() + getMessageGroupData() + loadMemberlist() + loadMessagegroupList() + setMemberStatus()

Abbildung 14: Nachrichtengruppenklasse

Die Registrierungsklasse:

Registration
- database : object - static instance : object + emailAdress : string + mail : string + tmpPassword : string
+ __construct() + static getInstance() + confirmationEmail() + createAccount() - generateTmpPassword() + register()

Abbildung 15: Registrierungsklasse

Die Systemnachrichtenklasse:

Abbildung 16: Systemnachrichtenklasse

Die Benutzerdatenklasse:

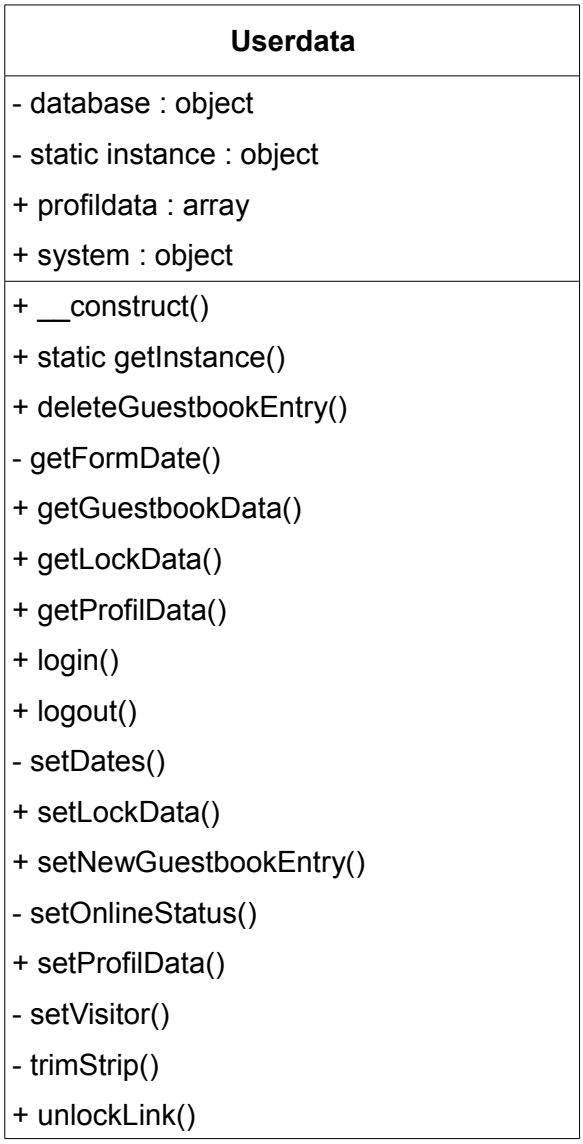


Abbildung 17: Benutzerdatenklasse

Die Validierungsklasse:

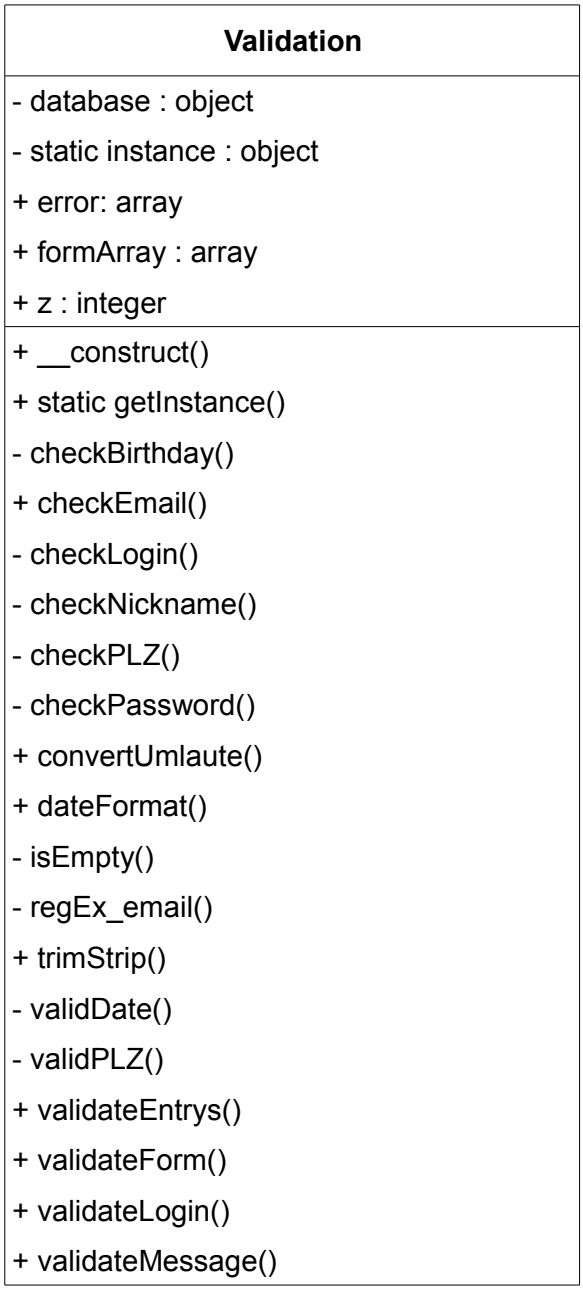


Abbildung 18: Validierungsklasse

Die Videoklasse:

Video
- database : object - static instance : object + youtube : object
+ __construct() + static getInstance() + getVideoData() + getVideoList() + setNewVideo() - splitData()

Abbildung 19: Videoklasse

Die Votingklasse:

Voting
- database : object - static instance : object
+ __construct() + static getInstance() + getVoting() + getVotingText() + setVoting()

Abbildung 20: Votingklasse

Die Wishlistklasse:

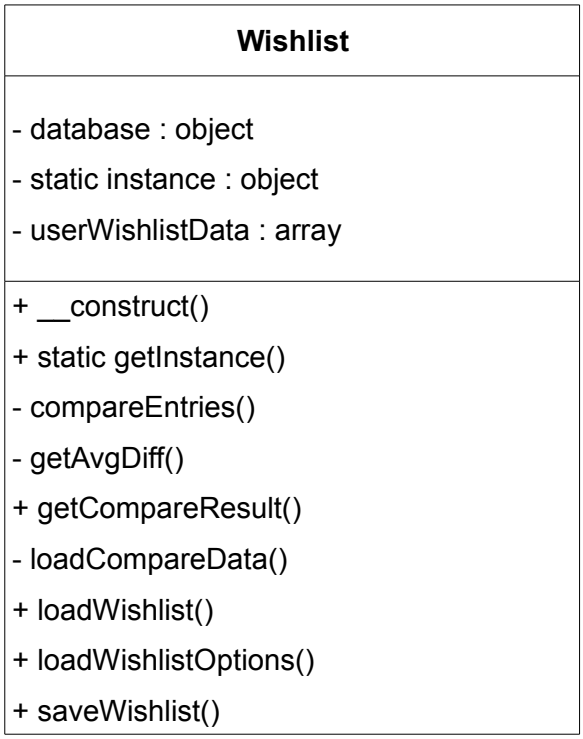


Abbildung 21: Wishlistklasse

Die XML-Klasse:

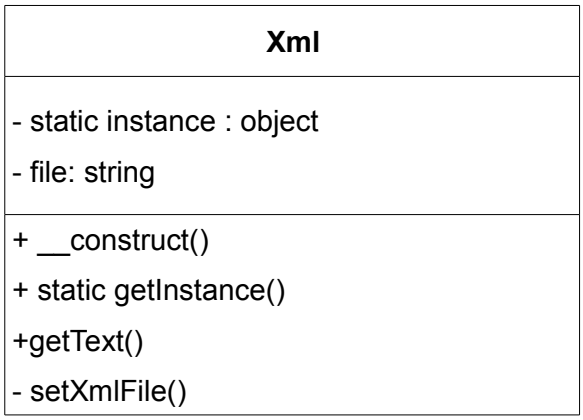


Abbildung 22: XML-Klasse

Die YouTubeklasse:

Youtube
- static instance : object
+ __construct() + static getInstance() + getTitle() +getVideoId() +getVideoInfos() - getXmlFile()

Abbildung 23: YouTubeklasse

4.3.2 Singleton-Pattern

Das Singleton-Pattern ist ein Erzeugungsmuster welches sich mit der Erzeugung von Objekten beschäftigt. Sein Vorteil liegt vor allem darin, dass die Anzahl der Objekte einer Klasse beschränkt wird, was zu einer Leistungssteigerung führen kann. Sinnvoll ist dieses Verfahren vor allem dann, wenn externe Ressource, wie zum Beispiel eine Datenbank, in einer Klasse gekapselt werden.

In der Datenbankklasse des Frameworks wird die Objekterzeugung über eine statische Methode mit dem Namen `getInstance()` bereitgestellt. Vorher muss natürlich noch die statische Variable `$instance` deklariert werden.

```
static private $instance = null;


static public function getInstance(){
      if(null === self::$instance){
            self::$instance = new self;
      }
      return self::$instance;
}
```

Das Singleton-Pattern findet jedoch nicht nur in der Datenbankklasse Anwendung sondern kann in jeder anderen Klasse verwendet werden, wo jeweils nur ein Objekt oder eine feste Anzahl von Objekten benötigt wird. [Bergmann, S.79]

4.4 Fehlerbehandlung - Exceptions

In größeren System ist es wichtig, eine Fehlerbehandlung durchzuführen. Fehler oder auch Ausnahmen (Exceptions) können prinzipiell überall auftreten, sei es bei der Verbindung zur Datenbank oder beim Umgang mit Dateien. PHP hält für die Fehlerbehandlung eine sogenannte Exception - Klasse bereit. Diese sieht im UML-Klassendiagramm so aus:

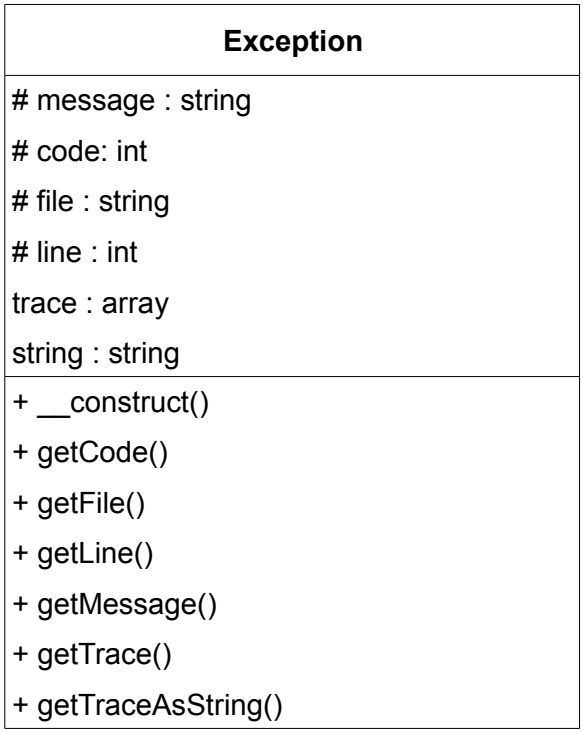


Abbildung 24: Exceptionklasse

[Bergmann, S.24]

Wird nun ein Code geschrieben, der für die Verarbeitung einer Ausnahme geplant ist, dann kann und wird dies überlicherweise in Klassen vorgenommen, die als Kind- von der Exception-Klasse abgeleitet sind. Die Methoden der zuletzt genannten können in den Kindklassen nicht neu definiert werden, da diese `final` sind. Mögliche Kindklassen können unter anderen eine `DB_Exception`-Klasse oder eine `File_Exception`-Klasse sein, das heißt eine solche, die alle Datenbankausnahmen, oder eine, die alle Dateiausnahmen kapselt. Prinzipiell sieht eine Kindklasse dann so aus (hier die `DB_Exception.class` als Beispiel)

```
<?php
	class DB_Exception extends Exception
	{
		privat function __construct($exception)
		{
			parent::__construct($exception);
```

```
            }
      }
?>
```

Durch das Schlüsselwort `extends` ist zu sehen, dass die Klasse `DB_Exception` eine Kindklasse von `Exception` ist. Damit die Kindklasse die Methoden der Exception-Klasse richtig nutzen kann, wird der Konstruktor (`__construct()`) der Elternklasse aufgerufen:

`parent::__construct($exception);`

[Krause, S. 156]

Um die Funktionsweise des Exceptionhandlings (Ausnahmenbehandlung) zu verdeutlichen, folgt hier ein Beispiel aus dem Quelltext der Datenbankklasse:

```
<?php
class Database{
   var $conn;
   private function connect(){
      try{
         $this->conn = mysql_connect(DB_HOST, DB_USER, DB_PW);
         if(!$this->conn)
         {
            throw new DB_Exception("Es konnte keine Verbindung zur Datenbank
            hergestellt werden.");
         }
         else
         {
            echo "Verbindung zur Datenbank wurde hergestellt";
         }
      }
      catch(DB_Exception $ex)
      {
         echo $ex->getMessage();
      }
}
?>
```

Das Abfangen eines Fehler geschieht mit Hilfe von `try-` und `catch`-Blöcken. Das eigentliche "Werfen" einer Ausnahme wird von `throw()` ausgeführt. Im Beispiel wird der "kritische" Bereich, also der, in dem Fehler auftreten können, von einem `try`-Block umschlossen. In diesem wird das Klassenattribut `conn` auf Gültigkeit überprüft. Ist `conn` gültig (`true`), wurde die Verbindung zur Datenbank hergestellt und eine entsprechende Meldung ausgegeben. Ist das Attribut ungültig (`false`), wird die Verarbeitung an der Stelle abgebrochen und eine Ausnahme geworfen (`throw`). Aufgefangen wird die Ausnahme danach durch den `catch`-Block. In diesem kann die weitere Verarbeitung des Fehlers, in diesem Fall die Ausgabe der Fehlernachricht, stattfinden. Hierbei ist zu beachten, dass der `throw`-Operator unabhängig von den `try-` und `catch`-Blöcken implementiert werden kann. Dadurch wird erst deutlich, wozu die Fehlerbehandlung in der Lage ist. Ein großer Vorteil in dieser Funktionsweise liegt nämlich darin, dass das Fehlerwerfen und -fangen mehr oder weniger unabhängig voneinander erfolgt: Geworfen wird der Fehler direkt an der Stelle, wo er auftritt. Dessen Behandlung jedoch kann an einer anderen Stelle vorgenommen werden. Folgende Übersicht fasst die Funktionsweise zusammen:

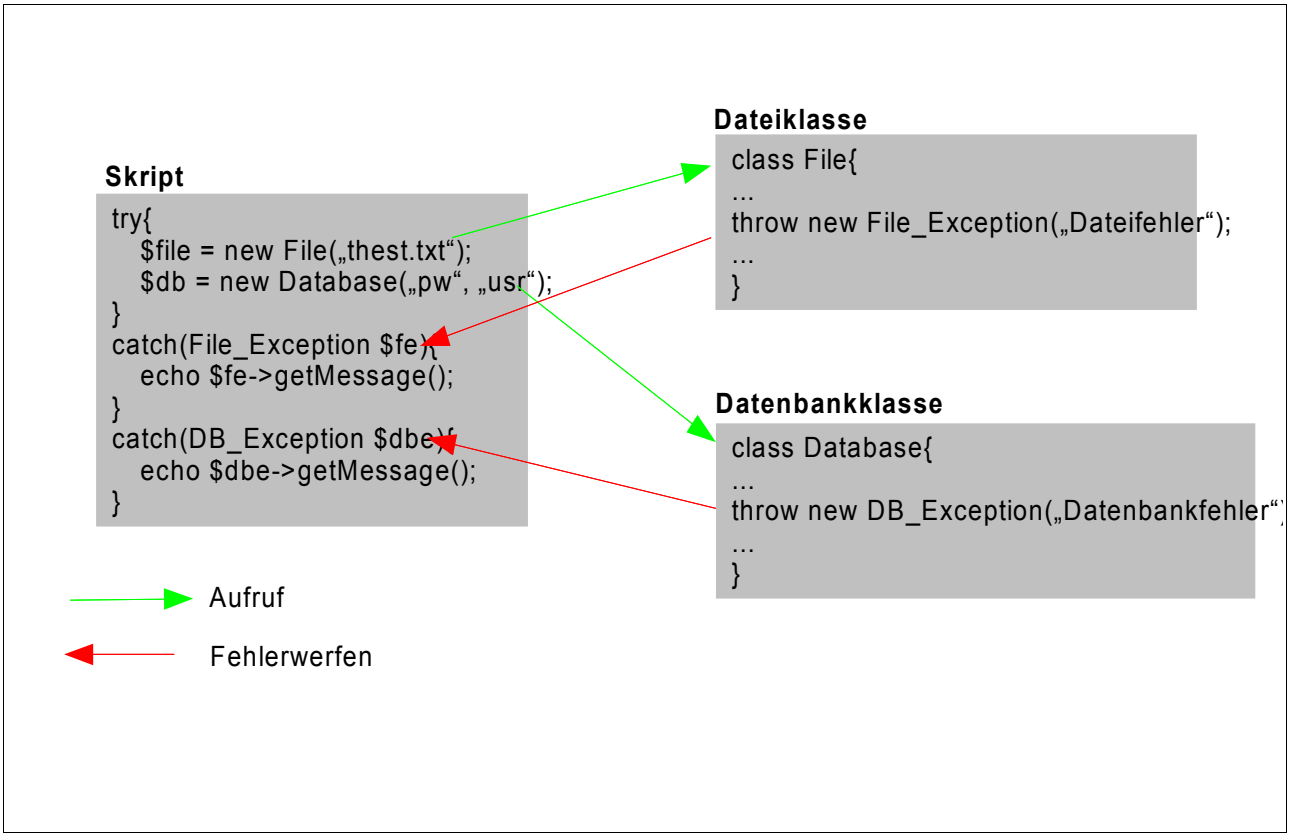


Abbildung 25: Exceptionhandling

4.5 Controller

Die Controller im MVC-Konzept übernehmen die wichtige Aufgabe, die Schichten Model und View, also Funktionalität und Design, so miteinander zu verbinden, dass am Ende ein funktionsfähiges System erstellt werden kann. Wie im Abschnitt 3.2 beschrieben teilt sich die Controller- Schicht in zwei Teile: den Router- und die Modul-Controller.

4.5.1 Router-Controller

Der wichtigste Controller des Frameworks ist der Router-Controller. Dessen Aufgaben sind vielfältig und von hoher Wichtigkeit. Zu ihnen gehören:

- Laden der Konfiguration

  Der erste Schritt, den der Router verarbeitet, ist das Laden der Konfiguration. Dazu gehört nicht nur das Bereitstellen der Systemparameter, sondern vor allem auch das Zur-Verfügung- Stellen der Objekte und Oberflächenparameter.

- Starten beziehungsweise anmelden der Session

  Der Vorgang des Startens und Anmeldens der Session wird über die PHP interne Funktion `session_start()` vorgenommen. Nähere Erläuterungen dazu erfolgen im Bereich 4.9 Session.

- `$_POST` Validierung

  Die Formulardatenvalidierung im Router ist nur der erste Schritt für eine Gesamtvalidierung. Übrprüft wird an dieser Stelle lediglich, ob alle Felder ausgefüllt sind oder nicht. Die $_POST-Validierung stellt letztlich ein Array bereit, in dem Informationen über das Ergebnis der Überprüfung stehen. In ihm sind dabei einzig die Formularfeldnamen angegeben, die bei der Validierung keine Werte enthielten. Das Ergebnisarray steht den Modulen zur Verfügung und kann in diesen ausgewertet werden.

- `$_GET` Validierung

  Die $_GET Validierung, also die Überprüfung der URL, stellt einen bedeutenden Faktor dar, um die Sicherheit für das System zu gewährleisten. Damit dies vollständig umgesetzt wird, ist es wichtig, dass keinerlei Code in irgendeiner Form ausgeführt werden kann. Das heißt also, dass die Hauptaufgaben dieser Validierung sein muss, HTML-Tags oder Datenbankabfragen sowie Schrägstriche, Leerzeichen und so weiter zu entfernen. Die genaue Überprüfung der GET-Variablen ist - wie schon bei der POST-Validierung - erst in den bestimmten Controllern möglich, da diese die benötigten Variablen kennen und daraufhin prüfen können.

- HTML-Head laden

  Der Headbereich einer HTML-Seite, also der Bereich, in dem der Titel der Seite, Stylesheetdateien, Skripte, Skriptdateien und Metaangaben deklariert sind, wäre viel zu überladen, wenn er statisch aufgebaut wäre. In Webandwendungen gilt jedoch, alles so schlank wie möglich zu gestalten. So ist es mehr als sinnvoll, den Headbereich dynamisch zusammenzusetzen, das heißt, Javascript und Stylesheetangaben zum Beispiel je nach Bereich hinzuzufügen.

- Seitenmenü laden

  Das Seitenmenü ist neben dem HTML-Footer der einzige Bereich, der statisch geladen wird. Dies passiert über die einfach Implementierung eines HTML-Templates. Zur besseren grafischen Umsetzung könnte an dieser Stelle das Menü dynamisch geladen werden. Dies ist allerdings nicht Teil der Anforderung und wird in der Arbeit nicht weiter besprochen.

- Modul und Seiteninhalt laden

  Die Modulauswahl ist die wichtigste Aufgabe des Routers, denn über die Module werden letztlich auch die Seiteninhalte bereitgestellt. Die Auswahl erfolgt über den GET-Parameter `mod`. Dieser wird neben der Angabe des Benutzernamens in jedem Link mitgesendet. In den Controllern, welche ausgewählt werden, findet die Datenmanipulation und die Einbindung der Designtemplates statt.

- HTML-Foot laden

  Der letzte Schritt für den Router ist es, die zusammengebaute HTML- Seite abzuschließen. Das heißt, die geöffneten HTML-Tags werden geschlossen und die Seite ist damit fertiggestellt.

Durch die Aufteilung im Router in die genannten Bereiche, wird eine Modularisierung ermöglicht, die es erlaubt, an dieser zentralen Stelle den Aufbau zu verändern. So bleibt nicht nur das Framework flexibel, sondern letztlich auch das Endergebnis für den Betrachter. Die folgende Abbildung fasst den Requestdurchlauf des Controllers zusammen:

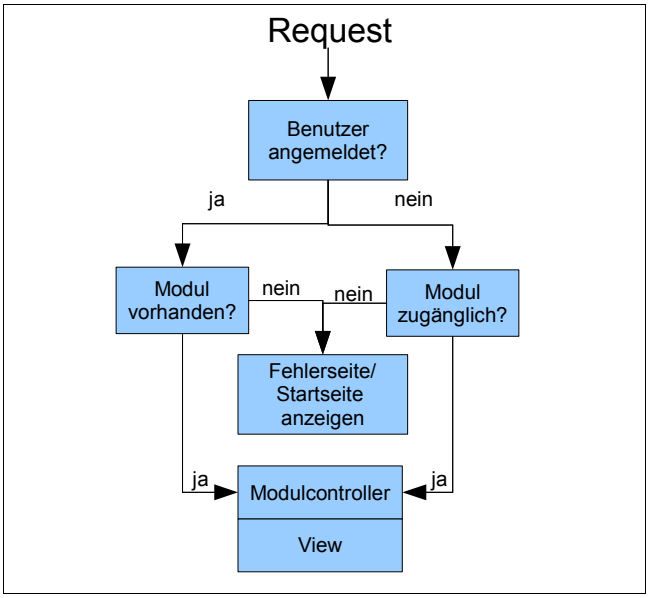


Abbildung 26: Routererablauf

4.5.2 Controllerablauf

In jedem Controller - abgesehen vom Router-Controller - ist der Ablauf gleich, denn immer müssen genau drei wichtige Arbeitsschritte erfolgen, die vom Controller beziehungsweise Skript abgeleistet werden.

Daten ermitteln → Daten verarbeiten → Daten aufbereiten und ausgeben

Das heißt, grundlegend muss natürlich erst einmal ermittelt werden, welche Aktionen der Benutzer ausgelöst hat. Es kann entweder sein, er hat auf einen Link geklickt oder ein Formular abgesendet. Im ersten Fall müssen lediglich Daten aus einer Datenbank oder Datei ausgelesen, aufbereitet und dann ausgegeben werden. Sollte jedoch der zweite Fall eingetreten sein, also das Absenden eines Formulars, dann müssen die Formulardaten erst einmal gespeichert werden, bevor dann wiederum Daten für die Anzeige gelesen, aufbereitet und ausgegeben werden können.

Im Skript sieht dieser Vorgang dann wie folgt aus:

```
<?php

if(isset($_POST)){
```

Es wurde ein Formular abgesendet. Das heißt, die Daten müssen erst einmal gespeichert oder in einer anderen Form verarbeitet werden. An dieser Stelle findet also die Weitergabe der Daten an das jeweils benötigte Objekt statt. In diesem selbst werden dann die Daten direkt verarbeitet, z.B. gespeichert, geändert.

```
}
```

Wurde kein Formular abgesendet oder sind die Daten bereits verarbeitet wurden, erfolgt an

dieser Stelle die Datenaufbereitung und dann mit

```
include(TEMPLATE_PATH."HalloWelt.tmpl.html");
```

wobei die Aufbereitung aus den Vorgängen des Datenauslesens und -vorbereitens besteht. Hierbei werden die benötigten Informationen meist aus der Datenbank oder einer Datei ausgelesen und dann für das Template in eine geeignete Form gebracht.

```
?>
```

4.6 Template

Eine Schicht des MVC-Konzeptes, auf dem das Framework aufbaut, ist die View-Schicht. In dieser geht es vor allem darum, die Daten zu präsentieren, die vorher im Model verarbeitet wurden. Es ist also wichtig, dynamischen Inhalt mit statischem zu verbinden, ohne dass zu viel Logik benutzt wird. Um diesen Anforderungen gerecht zu werden, hat es sich als erfolgreich erwiesen, sogenannte Templates zu verwenden. Im Framework selbst sind das Dateien, deren Inhalt fast nur aus HTML-Code besteht. Die dynamischen Elemente werde dann mithilfe von Platzhaltern eingefügt, die aus PHP-Code bestehen und meist IF-Verzweigungen, FOR-Schleifen und vor allem Datenausgaben enthalten. Programmlogik ist hingegen nicht enthalten, da diese der Controller- und Modellschicht vorbehalten ist. Hier nun ein Beispiel wie ein Template mit Platzhaltern aussehen kann:

```
<div>
  <?php foreach($resArr as $searchResult){ ?>
    <div class="" style="height:50px; margin-top:10px;">
      <div style="width:100px;float:left;">
        <?php if($searchResult['img']){ ?>
         <a href="?user=<?=$searchResult['userName']?>&mod=<?= MOD_PROFILE ?>">
              <img src="<?=$searchResult['img']?>"
                   alt="<?=$searchResult['userName']?>"
                   height="50" />
         </a>
        <?php }else{?>
          kein Bild
        <?php }?>
      </div>
      <div style="width:100px;float:left;">
        <a href="?user=<?=$searchResult['userName']?>&mod=<?= MOD_PROFILE ?>">
```

```
            <?= $searchResult['userName']?>
          </a>
        </div>
        <div style="width:100px;float:left;">
          <a href="?user=<?=$searchResult['userName']?>&mod=<?= MOD_PROFILE ?>">
            <?=$searchResult['name']?>
          </a>
        </div>
      </div>
      <?php  }?>
</div>
```

Die Platzhalter, also die Bereiche, in die der dynamische Inhalt eingefügt wird, sind von einer Template-Engine zur nächsten unterschiedlich. Für das Framework ist es am sinnvollsten die PHP eigene PHP-Template-Engine zu verwenden. Im Gegensatz zur sonst üblichen Smarty-Template-Engine entsteht bei der PHP- eigenen Lösung kein so großer Overhead durch die Extra -Zuweisung von Variablen und Daten.

Die o.g. Platzhalter der PHP-Variante stellen eine einfache Verfahrensweise dar, um den Inhalt einer Variablen auszugeben. Neben den öffnenden und schließenden Klammern mit Fragezeichen und "="-Zeichen(`<?= ?>`) muss lediglich die Variable angegeben werden, deren Wert angezeigt werden soll. Meist enthält ein Template allerdings nicht nur Ausgaben, sondern auch IF-Verzweigungen und FOR-Schleifen oder Ähnliches. Diese werden in normalen PHP-Klammern (`<? php ?>`) und am besten einzeilig geschrieben. Ein Vorteil in diesem Verfahren liegt darin, dass nichts extra installiert und somit auch keine zusätzliche Zeit investiert werden muss für die eventuelle Einarbeitung in die Template-Engine.

Bei der Erstellung der Templates muss großer Wert auf Wiederverwendbarkeit gelegt werden. Es ist zwar kaum möglich eins zu entwickeln, das universell einsetzbar ist. Dennoch sollte nicht für jedes Szenario genau ein spezifisches geschaffen werden. Um die Übersicht zu wahren, sollte also darauf geachtet werden, die Templates sinnvoll zu unterteilen. Es hat sich auch als nützlich erwiesen, neben den normalen Templates weitere zu besitzen, die Standardausgaben vornehmen können. So sollte es zum Beispiel für das Auftreten eines Fehlers oder für die Bestätigung eines Vorganges Standarttemplates geben, die je nach Anfrage eingesetzt werden.

4.7 MySQL-Datenbank

Die Datenbank zur Speicherung von Informationen und Daten enthält die im Kapitel Konzeption besprochenen Tabellen.

4.7.1 friends

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
user	int(255)	No			
friend	int(255)	No			
relation	varchar(20)	No	'Freund'		
confirmed	int(1)	No	'0'		

4.7.2 guestbook

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
owner	int(255)	No			
writer	int(255)	No			
writeTime	int(20)	No			
entry	varchar(3000)	Yes	NULL		
comment	varchar(2000)	Yes	NULL		

4.7.3 ignoreduser

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
user	int(255)	No			
ignored	int(255)	No			
typ	varchar(10)	No			

4.7.4 image_comments

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
imgId	int(255)	No			
userId	int(255)	No			
comment	varchar(255)	No			
commentTime	varchar(20)	No			

4.7.5 image_link

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
user	int(255)	No			
posx	int(5)	No			
posy	int(5)	No			
linkFrom	int(255)	No			
linkTime	int(20)	No			
imgId	int(255)	No			
unlocked	int(1)	No	'0'		

4.7.6 images

Name	Type	Null	Default	Extra	Comment
imageId	int(255)	No		auto_incre...	
user	int(255)	No			
album	varchar(255)	No			
image	varchar(200)	No			
imgName	varchar(255)	No			
unlocked	varchar(1)	No			

4.7.7 locksettings

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
user	int(255)	No			
onlinestatus	int(1)	No			
guestbook	int(1)	No			
profil	int(1)	No			
profilview	int(1)	No			
galery	int(1)	No			
imagelinks	int(1)	No			
wishlist	varchar(1500)	No			
videos	int(1)	No	'0'		Wenn dieser Wert ...
locktype	varchar(10)	No	'no'		

4.7.8 member

Name	Type	Null	Default	Extra	Comment
ID	int(11)	No		auto_incre...	
email	varchar(100)	No			
passw	varchar(50)	Yes	NULL		
birthday	date	Yes	NULL		
localID	varchar(5)	Yes	NULL		
activated	int(1)	No	'0'		
registrationDate	datetime	Yes	NULL		

4.7.9 messagegroups

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
user	int(255)	No			
member	int(255)	No			
groupName	varchar(50)	No			

4.7.10 messages

Name	Type	Null	Default	Extra	Comment
messageId	int(255)	No		auto_incre...	
sender	varchar(5)	No			
acceptor	varchar(100)	Yes	NULL		
title	varchar(100)	Yes	NULL		
message	varchar(1000)	Yes	NULL		
archived	int(1)	No			
deleted_acceptor	int(1)	No	'0'		
deleted_sender	int(1)	No	'0'		
lastAction	timestamp	No	CURRENT_TIMEST...		
groupMessageId	int(1)	Yes	NULL		
readStatus	int(1)	No	'0'		
blinddate	int(1)	Yes	'0'		
requestMessage	int(1)	No	'0'		Wenn 1 dann Nachr...

4.7.11 profil

Name	Type	Null	Default	Extra	Comment
userID	int(255)	No	'0'		
gender	varchar(1)	Yes	NULL		
name	varchar(100)	Yes	NULL		
forename	varchar(100)	Yes	NULL		
city	varchar(50)	Yes	NULL		
height	varchar(3)	Yes	NULL		
weight	varchar(3)	Yes	NULL		
hairClr	varchar(15)	Yes	NULL		
eyeClr	varchar(15)	Yes	NULL		
smoke	varchar(20)	Yes	NULL		
homepage	varchar(100)	Yes	NULL		
work	varchar(50)	Yes	NULL		
lookFor	varchar(20)	Yes	NULL		
userName	varchar(10)	Yes	NULL		
lastAction	timestamp	No	'0000-00-00 00:00...	on update ...	
online	tinyint(1)	No	'0'		
emailmessages	int(1)	No	'1'		
systemmessages	int(1)	No	'1'		
img	varchar(100)	Yes	'design/img/syste...		
friends	varchar(10000)	Yes	NULL		
statusText	varchar(500)	Yes	NULL		
visitors	varchar(3000)	Yes	NULL		
wishlist	varchar(3000)	Yes	NULL		
wishlist_self	varchar(3000)	Yes	NULL		
pause	int(1)	No	'0'		

4.7.12 system_messages

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
user	int(255)	No			
message	varchar(500)	No			
messageTime	int(20)	No			
readStatus	int(1)	No			
modul	varchar(30)	No			
link	varchar(400)	Yes	NULL		

4.7.13 unlocked_user

Name	Type	Null	Default	Extra	Comment
user	int(255)	No			Id des Benutzers
unlocked_userid	int(255)	No			Id des freigeschalte...
requestMessage	varchar(250)	No			
onlinestatus	int(1)	No	'0'		1 Wenn onlinestatu...
guestbook	int(1)	No	'0'		1 wenn GB gesehen...
profil	int(1)	No	'0'		1 Wenn Profil eiges...
imagelinks	int(1)	No	'0'		1 Wenn Benutzer v...
galery	int(1)	No	'0'		1 Wenn Freigeschal...
videos	int(1)	No	'0'		Wenn der Wert auf...
confirmed	int(1)	No	'0'		User ist erst freiges...
lastAction	timestamp	No	CURRENT_TIMEST...	on update ...	

4.7.14 video

Name	Type	Null	Default	Extra	Comment
id	int(10)	No		auto_incre...	
user	int(10)	No			
videoId	varchar(15)	No			
videoTitle	varchar(150)	Yes	NULL		
uploadedTime	int(20)	No			

4.7.15 video_comments

Name	Type	Null	Default	Extra	Comment
id	int(255)	No		auto_incre...	
videoId	int(255)	No			
userId	int(255)	No			
comment	varchar(255)	No			
commentTime	varchar(20)	No			

4.7.16 voting

Name	Type	Null	Default	Extra	Comment
userId	int(255)	No			
voterSex	varchar(1)	No			
voteValue	varchar(1)	No			
voter	int(255)	No			

4.8 jQuery

An einigen Stellen im Framework ist es sehr von Nutzen, wenn DOM-Manipulationen oder asynchrone Anfragen an den Webserver vorgenommen werden können. Das Javascript-Framework jQuery hält für diese Anforderungen komfortable Funktionen bereit. Weitere Funktionen und Möglichkeiten von jQuery sind:

- DOM-Selektierung und -Navigation mit Hilfe von CSS und XPath,
- Erweitertes Event-System,
- Effekte und Animationen,
- Beliebige Erweiterbarkeit.

Der Vorteil im Einsatz von jQuery liegt aber vor allem darin, dass Änderungen am Verhalten oder am Ablauf nicht wie im normalen Javascript an mehreren Stellen vorgenommen werden müssen, sondern an einer zentralen, im Head-Bereich der Seite[Resig].

Ein typischer jQuery-Codeabschnitt sieht wie folgt aus:

```
$("#image img").live("mouseover", function(){
	$(this).css("cursor", "crosshair");
	return false;
});
```

Mit dem "$"-Operator wird dabei ein Objekt gebildet auf das dann mit verschiedenen Funktionen, im dem Fall mit "live", zugegriffen werden kann. Zu den Funktionen zählen solche die zum Beispiel die DOM-Struktur manipulieren, Events abfangen und verarbeiten, Effekte ausführen oder Ajax-Anfragen absenden können. Gerade im Bereich der Ajax-Funktionalitäten erspart das jQuery-Framework dem Programmierer sehr viel Arbeit in dem es zum Beispiel die Bildung des benötigten XMLHttpRequest-Objektes übernimmt. Eine Ajax-Anfrage abzusenden sieht bei jQuery demnach so aus:

```
$.post("linkuser.php",
	{user:user, imgTitle:imgTitle},
	function(res){
		alert(res);
	},
	"html"
);
```

Es werden nur maximal vier Angaben benötigt:

- Zielskript (im Beispiel "`linkuser.php`") (Benötigt),

- [optional] Daten, die an das Zielskript übergeben werden sollen (`user:user, imgTitle:imgTitle`)
- [optional] eine Funktion, die die Verwertung des Ergebnisses übernimmt (in dem Fall eine anonyme Funktion: `function(res){...}`)
- [optional] Datentyp des Ergebnisses (`html`, möglich ist aber auch xml oder text)

Statt `$.post` kann auch `$.get` oder `$.ajax` geschrieben werden. In letzterem Fall (`$.ajax`) muss allerdings, als Parameter, die Übermittlunsgmethode (POST oder GET) mit angegeben werden.

4.9 Session

Ein wichtiges Element des Frameworks ist der Bereich, in dem der Benutzer sich an der Anwendung an- und abmeldet. Nur durch dieses Anmeldeverfahren ist es möglich, dass jedem ein eigener Bereich zur Verfügung gestellt werden kann. Ohne Anmeldung gäbe es keine eindeutige Identifikation und somit auch keine ordnungsgemäße Zuordnung von Informationen.

Das grundlegende Problem bei einer Benutzeranmeldung und bei der Datenhaltung über mehrere Seiten liegt darin, dass das dafür verwendete Protokoll (HTTP) zustandslos ist. Der Webserver übernimmt die Daten einer Anfrage, übergibt sie dem verarbeitenden Skript und "vergisst" anschließend die Anfrage wieder und damit auch die Daten. Wenn diese allerdings über mehrere Anfragen hinweg bekannt sein sollen, dann ist es wichtig, diese Daten irgendwo zwischenzuspeichern, ohne den Weg über eine Datenbank zu wählen. Es gibt folgende Möglichkeiten, dass Benutzerdaten über einen ganzen Vorgang (mehrere Anfragen) zur Verfügung stehen:

- Benutzerdaten per URL mitführen

  Das würde bedeuten, dass die Benutzerdaten an die URL angehängt werden und somit für jeden sichtbar sind, was im Falle einer verschlüsselten Nachricht sehr ungünstig wäre. Ein weiterer Nachteil dieser Methode liegt in der Begrenztheit einer URL auf maximal 2000 Zeichen.

- Benutzerdaten per Cookies mitführen

  Die Speicherung von Informationen per Cookie bedeutet, dass diese in einer kleinen Datei (Cookie) beim Client gespeichert werden. Es kann allerdings sein, dass Cookies vom Client beziehungsweise vom Browser nicht akzeptiert werden, wodurch dann das System nicht funktionieren würde.

- Benutzerdaten per Session mitführen

> Die Verwendung von Sessions, "Sitzungen", bietet vor allem den Vorteil, dass nicht bei jedem HTTP-Request, also jeder Anfrage vom Client (Browser) an den Webserver, alle Daten und Informationen mitgesendet werden. Dadurch ergibt sich noch ein weiterer Vorteil: Da die Daten nicht transportiert werden, sondern nur auf dem Webserver liegen, ist die Gefahr eines Angriffs oder von Manipulation während der Datenübertragung wenig bis gar nicht gegeben.
>
> [Krause, S. 225]

Sessions funktionieren grundlegend so, dass jedem Browser eine eindeutige Identifikation gegeben wird, eine sogenannte Session-ID. Mit dieser Kennung ist es möglich, auf Sessionvariablen auf dem Server zuzugreifen. In solchen können dann Werte und Informationen gespeichert werden. Das folgende Bild verdeutlicht den Aufbau und Einsatz eines Systems, in dem Sessions verwendet werden.

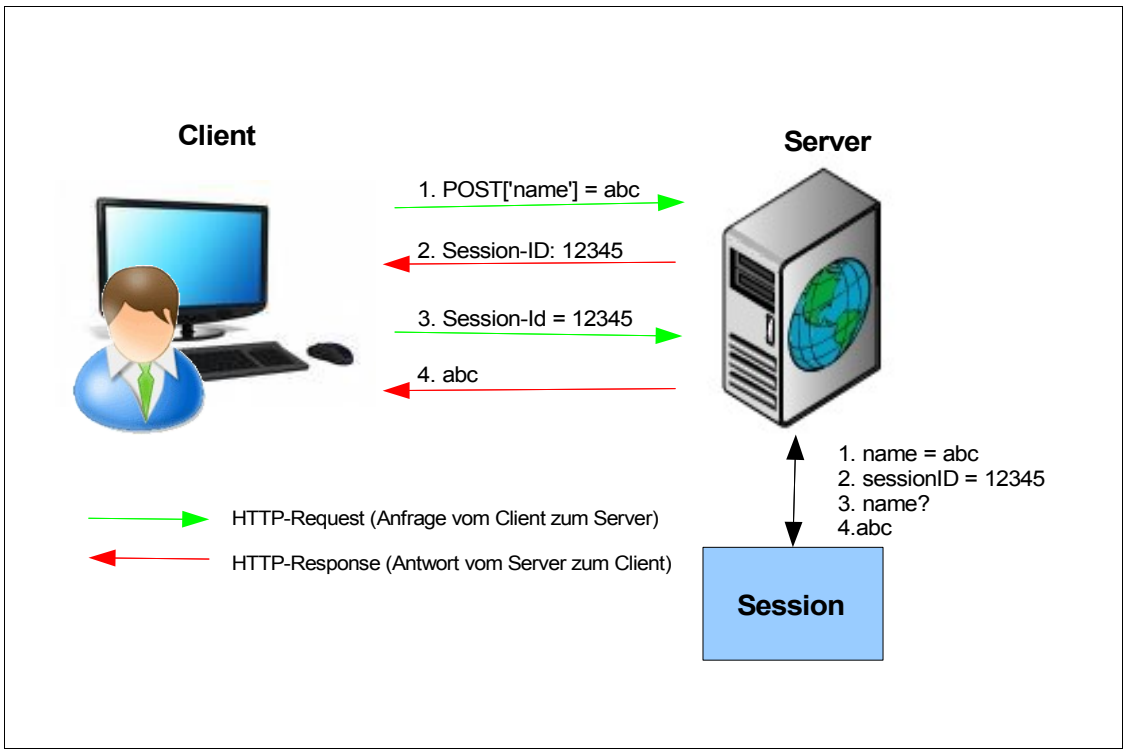


Abbildung 27: Sessiondarstellung

Zu sehen sind die unabhängigen, weil zustandslosen HTTP-Aktionen, Request und Reponse, die vor allem per GET und POST ausgelöst werden. Die übertragenen Daten können nun entweder in einer Datenbank gespeichert werden oder eben in der Session als Sessionvariablen. Für den schnellen und trafficarmen Datenverkehr sollte man letzterer Vorgehensweise den Vorzug geben, da somit erst der Aufbau einer Verbindung von und zur Datenbank zum Speichern und dann einer zum Lesen der Daten entfällt. Stattdessen stehen hier die Daten für die Dauer der Session auf dem Webserver zur Verfügung.

Der Zugriff auf Daten einer Session kann dann wie folgt aussehen:

```
<?php
```

```
session_start();
```

Bevor auf eine Session zugegriffen werden kann, muss diese natürlich erst einmal mit der Funktion `session_start()` initiiert werden. Ist das bereits der Fall, das heißt wurde die Session bereits gestartet, wird sie mit dieser Funktion erneut aufgenommen.

```
$_SESSION["userName"] = "Max";
```

Die Session speichert nur in einem Array jeden beliebigen Wert. Dabei sollte jedoch darauf geachtet werden, dass die Werte vorher validiert werden, wenn sie per POST oder GET gesendet wurden.

```
echo $_SESSION["userName"];
```

Der Zugriff auf Sessiondaten erfolgt dann über den jeweiligen Namen der Variablen. In dem Beispiel wird "Max" ausgegeben.

```
?>
```

Im Framework werden Sessions vor allem dazu verwendet, Benutzerdaten zu halten, während dieser am System angemeldet ist. Dazu gehören unter anderem der Benutzername und die Benutzerkennung, das heißt die Benutzer-ID. Während des Abmeldevorganges wird die Session gelöscht wodurch die Daten für andere nicht mehr einsehbar sind.

4.10 URL-Rewriting

Eine Anforderung aus dem Lastenheft ist es, dass das Framework mit sprechenden URLs arbeitet. Grundlegend bedeutet URL-Rewriting nichts anderes, als dass die Parameter in einer URL nach einer bestimmten Vorschrift umgeschrieben werden und damit für den Benutzer auch leichter lesbar sind. Hier ein Beispiel:

aus: www.beispiel.de/index.php?user=Max&action=video&title=beispiel_video

kann durch die Rewrite-Engine des (Apache) Webservers folgende URL werden:

www.beispiel.de/Max/video/beispiel_video

Die Vorschrift für diese Umwandlung der URL kann in der .htaccess Datei des Servers vorgenommen und vom Rewriting Modul (mod_rewrite) des Apaches Webservers verarbeitet werden. Auf das Beispiel bezogen sieht das URL-Rewriting wie folgt aus:

```
RewriteEngine on
```

```
RewriteRule ^/(.*)/(.*)/(.*)$ /index.php?user=$1&action=$2&title=$3
```

Die Vorschrift ist dabei grundlegend in zwei Teile gegliedert. Der erste - zwischen `^` und `$` – repräsentiert dabei die URL, bei der die "dynamischen" Bereiche durch reguläre Ausdrücke

dargestellt werden. Der danach folgende zweite Teil stellt die URL dar, wie sie dann vom Framework verwendet wird. Die URL, die der Benutzer sieht, bleibt hingegen gleich. Die in Klammern stehenden regulären Ausdrücke "`(.*)`" werden in der Variablen "`$`" gespeichert. Dabei gilt, der reguläre Ausdruck, der zuerst gefunden wurde, wird in `$1` gespeichert, der zweite in `$2` und so weiter. [Wetsch(1)]

Mit diesem Verfahren ist es dem Benutzer möglich die URL besser zu verstehen und sich dadurch zu merken. Das URL-Rewriting wird aber vor allem bei der Suchmaschinenoptimierung verwendet um Suchmaschinenfreundlichere URLs zu erstellen. Weitere Vorteile bei der Verwendung von mod_rewrite sind:

- Spambots (Guestbook-Spam, Blog-Spam, Foren-Spam) können ausgeschlossen, d.h. eingedämmt oder ganz verhindert, werden .
- Schutz vor "Hackern" und/oder "Script-Kiddies" wird gewährt.
- Content-Klau wird verhindert.
- Weiterleitung oder Sperrung wird ermöglicht nach bestimmten Client-Kriterien, wie z.B. Referer, Client, IP-Adresse etc.
- Content-Regeneration bei Umstrukturierung der Webseite(n),
- zeitabhängige Weiterleitungen,
- dynamische Mirrors.

[Wetsch(2)]

4.11 Dokumentation (PHPDoc)

Neben einem guten Programmierstil und einer ausgiebigen Testphase ist es vor allem die Dokumentation des Programmcodes, die den Wert des Frameworks und somit die Anzahl der Einsätze steigert. Ulf Wedel hat mit PHPDoc im Jahr 2000 einen ersten Standard beziehungsweise Ansatz, auf der Basis von Javadoc, vorgestellt, mit dem es möglich ist PHP-Code umfassend zu dokumentieren. Die Stärke von PHPDoc liegt vor allem darin, eine übersichtliche Dokumentation von objektorientiertem Programmcode zu gestalten. Die geschieht vor durch die Dokumentationssprache und durch das PHP-Programm phpDocumentor. Damit lassen sich HTML-Übersichten über Klassen und deren Attribute sowie Methoden erstellen. Im Quelltext einer Klasse kann die Dokumentation dann wie folgt aussehen:

```
/**
*setIgnoreStatus-Methode
*@param string $user Name des Benutzers
*@param string $ignoreFriend Name desjenigen der gesperrt werden soll
*@param string $ignoreOption Wert der Ignorierung
*@return array Gibt ein Array zurück mit einem Freundedaten Array und der Anzahl
```

```
      der Freunde
*/
public function setIgnoreStatus($userId, $ignoreFriendId, $ignoreOption){
…
}
```

Der Kommentarblock mit PHPDoc beginnt immer mit "`/**`" und endet mit "`*/`". In diesem Block können dann über die bestimmte Schlüsselwörter unter anderen übergebene Parameter beschrieben (im Beispiel oben: "`@param ...`") oder Rückgabewerte angegeben werden ("`@return...`"). Mehr zu dieser Möglichkeit der Quelltextdokumentation ist unter www.phpdoc.org oder bei [Eichborn] zu finden.

4.12 Sicherheit

In einem Framework und speziell in der Anwendung, die mit diesem erstellt wird, und auch dann, wenn man mit Benutzerdaten umgeht, steht die Sicherheit an oberster Stelle. Denn Benutzerdaten wie Passwort oder E-Mailadressen sollten nicht so einfach zugänglich sein. Im Folgenden werden einige Angriffsmöglichkeiten besprochen und wie das System davor geschützt werden kann.

4.12.1 SQL-Injection

Bei dieser Angriffsart sendet der Angreifer SQL-Befehle über ein Formular oder eine URL an den Server und versucht somit an Benutzerdaten heranzukommen. Wenn diese SQL-Befehle nicht abgefangen werden und somit ungefiltert in SQL-Abfragen Eingang finden, können Daten ausgelesen und verändert werden.

Um diesem Angriff nicht zu zulassen, werden sämtliche Eingaben über POST oder GET vorher validiert, bevor sie weiterverarbeitet werden. Die Validierung kann mit der PHP-Funktion `mysql_real_escape_string()` und einer gesonderten Überprüfung erfolgen, in der der Wert auf Typ und Beschaffenheit kontrolliert wird. Dies setzt allerdings auch voraus, dass die Validierung weiß welcher Typ an der Stelle erwartet wird. Eine weitere Methode ist es, den übermittelten Wert , daraufhin zu prüfen, ob er bestimmte SQL-Befehle enthält und diese dann entweder herauszufiltern oder den Wert ganz zu löschen. Grundsätzlich sollte darauf geachtet werden, dass nicht ausschließlich eine Methode verwendet wird. In der Validierungsklasse gibt es für diesen Angriff folgende Methode:

```
1 public function trimStrip($msg){
2     $msg = trim($msg);//Alle Leerzeichen vor und hinter dem String entfernen
3     $msg = htmlentities($msg, ENT_QUOTES, "UTF-8");
4     $msg = strip_tags($msg);      //HTML-Tags entfernen
```

```
5     $msg = mysql_real_escape_string($msg);
6     return $msg;
7 }
```

Diese Methode filtert nicht nur MySQL-Befehle aus den POST oder GET Variablen (Zeile 5), sondern auch HTML-Befehle/Tags (Zeile 4). Zeile 3 des Quelltextes codiert HTML-Tags in das UTF-8 Format.

4.12.2 Cross-Site Scripting

Beim Cross-Site Scripting (XSS) wird von Angreifern Schadcode auf den Server geschleust. Dies kann entweder über die URL, über Formulare oder Einträge in eine Datenbank, zum Beispiel bei einem Gästebuch, geschehen. Der Schadcode kann entweder mit HTML-Tags, Javascript oder sogar PHP-Code versehen sein. Wenn der Code ungefiltert gelesen oder sogar ausgeführt wird, kann es dazu kommen dass Daten vom Server oder auch vom Benutzer ausgelesen werden.

Der Schutz vor XSS sieht so aus: Jede Eingabe, die der Benutzer vornimmt, wird gefiltert anhand von Blacklisten, Listen mit verbotenen Wörtern, oder Whitelisten, Wörtern, die nur erlaubt sind. PHP selbst hat mit *htmlspecialchars()* eine Funktion , die HTML-Sonderzeichen filtern kann. An dieser Stelle folgt der Verweis auf den Beispielquelltext, wie er schon im Bereich SQL-Injection angezeigt wurde. Dabei wird statt *htmlspecialchars()* die Funktion htmlentities() verwendet, die jedoch ein ähnliche Arbeitsweise hat. Die Funktion htmlentities() wandelt alle HTML-Zeichen um und die Funktion *htmlspecialchars()* nur bestimmte Symbole und Zeichen (&, ", ', <, >).

4.12.3 Session-Hijacking

Das Session-Hijacking versucht eine Schwachstelle bei der Verwendung von Sessions aus zu nutzen, die darin besteht, dass der Angreifer versucht, die Session-ID eines Benutzers herauszufiltern und somit in die bestehende Session einzugreifen. An die Session-ID gelangt der Angreifer entweder über die URL, wenn die Session-ID an die URL gehangen wird, oder über die Cookies des Clients. Die URL oder Cookies kann der Angreifer dabei vor allem durch das Einschleusen von Javascript-Code auslesen.

Um zu verhindern, dass Angreifer an die Session-ID gelangt, ist das Framework so gebaut, dass die Session-ID nicht mit der URL mitgesendet wird. Zusätzlich werden jegliche Javascript-Bestandteile aus den POST oder GET Anfragen gefiltert. Abschließend wird beim Abmelden des Benutzers die Session vollständig gelöscht, so dass keinerlei Möglichkeit des Auslesens besteht. [Codecasters]

4.12.4 Cross-Site Request Forgery

Dieses Angriffsverfahren basiert darauf, dass der Hacker, der Angreifer, in eine bestehende Session eingreift (XSS oder ähnliches) und so den Benutzer der Session eine vom Angreifer vorgefertigte URL auf dem Server ausführen lässt. Ziel dieses Verfahrens ist es, Daten zu manipulieren um so, in weiteren Schritten Zugang zum System zu erhalten sowie Daten auslesen zu können.

Schutz vor diesem Angriff ist eine Methode, die die Verbindung von Webserver zum Browser eindeutig identifiziert. Dazu muss eine Schlüssel oder Token erstellt werden, der nur dem Webserver und dem Browser bekannt ist. Da der zufällig generierte Schlüssel dem Angreifer vorher nicht bekannt ist, funktionieren seine präperierten Links nicht mehr. Es hat sich bewährt, dass diese Schlüssel mindestens acht Stellen haben. Damit ist das Opfer weitestgehend auch vor sogenannten Brute-Force-Attacken sicher. Im Framework wird zum Schutz als erstes der Schutzmechanismus aufgerufen wie er schon beim XSS eingesetzt wird.[Bachfeld]

4.12.5 Directory Traversal

Directory Traversal ist eine Angriffsmethode, bei der Pfadangaben von Webservern so verändert worden sind, dass auf sensible Dateien und Ordner zugegriffen werden kann. Dies könnte letztlich darin münden, dass Informationen wie Passwörter oder ähnliches preisgegeben werden. Wird zum Beispiel in einer URL ein Dateiname übergeben ("www.beispiel.de/dateiladen.php?datei=halloWelt.html"), kann diese Schwachstelle ausgenutzt werden, indem der Dateiname mit einer Pfadangabe (zum Beispiel "../../passwort.html") ausgetauscht wird. So kann auf Dateien und Programme zu gegriffen werden.

Verhindern kann man diesen Angriff, wenn die GET-Parameter vor der Ausführung auf Pfadangaben geprüft und diese dann noch durch eine URL-Kodierung entschärft werden. Das Framework ist vor diesem Angriff geschützt, da keine Dateinamen oder Pfade über die URL übergeben werden.

5. Funktionstests

Die in diesem Kapitel beschriebenen Testszenarien werden als Systemtest im Black- Box-Testverfahren nach Fertigstellung des Frameworks durchgeführt. Bei diesem Verfahren testen externe Personen das System. „Extern“ bedeutet in diesem Sinne, dass die Personen mit der Entwicklung nichts zu tun hatten und damit auch die Funktionsweise des Systems nicht kennen, deshalb „Black Box“ genannt. Ziel dieser Tests ist es, Auswirkungen von System- beziehungsweise Programmierfehlern hervorzurufen, um somit Rückschlüsse auf deren Ursache zu ziehen und sie letztlich beseitigen zu können. Die Testszenarien stellen ausschließlich Funktionstests auf neutralen, also nicht gestalteten Oberflächen dar, da im vorliegenden Fall lediglich ein Framework, ein Funktionsgrundgerüst, und keine Anwendung getestet wird.

5.1 Registrieren und Anmelden

Folgende Schritte muss eine Testperson bei der Evaluierung der Registrierung durchlaufen:

1. E-Mailadresse eingeben

2. Registrierungsemail empfangen und auf den Registrierungslink klicken

3. geforderte Daten ausfüllen (Passwort, Benutzername, Geburtsdatum, Postleitzahl)

Die Anmeldung setzt dagegen auf folgenden Ablauf bei der Evaluierung:

1. E-Mailadresse eingeben

2. Passwort eingeben

3. auf Login klicken

Der Vorgang des Abmeldens erfolgt über den Menüpunkt "ausloggen".

5.2 Profil ausfüllen/Profilfoto

Nachdem der Tester sich erfolgreich am System anmelden konnte, hat er die Aufgabe, seinen Profilbereich zu finden und seine Daten dort einzugeben. Dazu gehört nicht nur das Ausfüllen der Felder, sondern auch das Erstellen eines Profilfotos. Um das zu bewältigen, muss der Tester im persönlichen Bereich des Systems ein Bild hochladen, im Anschluss daran einen Abschnitt des Bildes auswählen und den Vorgang mit dem Klick auf "speichern" abschließen.

5.3 Videos

In diesem Bereich, hat der Tester die Aufgabe, Videos von der Onlinevideoplattform YouTube zu seiner Galerie hinzuzufügen. Dazu nimmt er den Link des Videos und trägt ihn in das dafür vorgesehene Feld ein. Um den Vorgang abzuschließen, muss er das Formular nur absenden und

danach ist das Video Bestandteil der Galerie.

Eine weitere Aufgabe der Testperson ist es, ein Video aus seiner Videogalerie auszuwählen und dazu einen Kommentar zu verfassen.

5.4 Bilder

Folgende Arbeitsschritte muss der Tester zur Durchführung dieses Test bewältigen:

1. im Menüpunkt Bilder ein Album anlegen

2. diesem Album Bilder hinzufügen

3. vereinzelte Bilder auswählen und deren Namen ändern sowie Kommentare und Verlinkungen hinzufügen.

5.5 Freunde suchen und hinzufügen

Über die Suchfunktion muss ein Benutzer gefunden werden, dem nach der Besichtigung des Profils ein Antrag auf Freundschaft gesendet werden soll. Nach der Bestätigung des Antrags hat der Tester nun den Benutzer in der Freundesliste zu finden.

5.6 Nachrichten an Freunde schicken

Über den Menüpunkt Nachrichten soll eine Nachricht an zwei Freunde gesendet werden. Einmal über die Auswahlliste und dann über die direkte Eingabe des Benutzernamens.

5.7 Nachrichten lesen und beantworten, löschen

Unter dem Menüpunkt Nachrichten müssen durch die Testperson Nachrichten empfangen (gelesen) und beantwortet werden. Dazu muss die testende Person auf die entsprechenden Links auf der Oberfläche klicken. Zum Löschen einer Nachricht, reicht es aus auf den Link "löschen" zu klicken.

5.8 Bereiche sperren

In der Sperrverwaltung, die über den Menüpunkt "Einstellungen" zu erreichen ist, muss die Testperson bestimmte Bereiche auswählen, die der Öffentlichkeit nicht mehr angezeigt werden sollen. Diese Bereiche sind dann von anderen Testern auf Ihre Sperrung hin zu überprüfen. Wird ein gesperrter Bereich durch die Testperson betreten, soll diese den Antrag auf Freischaltung stellen, der von dem Benutzer, an den der Antrag gestellt wurde, verwaltet wird.

5.9 Freunde ignorieren

Die Testperson soll für diesen Test die Profilseite eines Freundes aufrufen und dann auf den Link "ignorieren" klicken. In der nächsten Übersicht, werden der Testperson drei Möglichkeiten geboten, von denen sie sich eine aussuchen kann. Bestätigt wird die Einstellung mit dem Klick auf den "speichern"-Knopf. Die Testperson kann den Vorgang jedoch auch abbrechen, indem sie auf den Link mit dem Profilnamen des Freundes klickt.

5.10 Gästebucheintrag vornehmen

Im Menübereich Gästebuch soll ein Testbenutzer einen Eintrag verfassen und diesen mit "speichern" bestätigen. Der Autor eines Eintrages und der Besitzer des Gästebuchs haben dann noch die Möglichkeit, die jeweiligen Einträge zu löschen.

5.11 Blinddate finden

Das Blinddate, also eine Konversation mit einem anderem Benutzer, den die Testperson nicht kennt, kann der Tester über die Suche aktivieren. Damit wird ihm die Möglichkeit geboten, eine Nachricht an eine unbekannte Person zu versenden und sich dann mit dieser zu unterhalten.

6. Fazit

6.1 Zusammenfassung

In der vorliegenden Arbeit wurde der Prozess der Konzeption und Entwicklung eines Frameworks für ein Communitynetzwerk dargestellt. Angefangen wurde mit allgemeinen Betrachtungen zu Communitynetzwerken und Frameworks. Im nächsten Schritt wurden die Anforderungen und Aufgaben der Problemstellung in einem Pflichtenheft dargestellt. Dabei wurde unter anderem auf die Zielbestimmungen, Produkteinsatz und -umgebung sowie auf die Funktionen eingegangen. Der dritte und auch wichtigste Abschnitt dieser Arbeit beschäftigte sich mit dem Entwurf und der Konzeption des zu entwickelnden Frameworks. In diesem Abschnitt wurde das Entwurfsmuster, die Klassen und die Datenbank sowie die Konfiguration geplant und besprochen.
Der letzte große Abschnitt der Arbeit, die Implementierung, beschäftigte sich im Anschluss mit der Entwicklung des Frameworks und ging vor allem auf die Klassen, die einzelnen Schichten des Entwurfsmusters, die Fehlerbehandlung und die Umsetzung der Datenbankplanung ein. Weitere Punkte in diesem Kapitel waren unter anderen auch die Sessionverwaltung, die Dokumentation und vor allem die Aspekte der Sicherheit des Frameworks. Hierbei wurden Angriffsmöglichkeiten genannt und beschrieben sowie Schutzmechanismen und deren Einsatz im Framework erläutert.
Schließlich wurde im fünften Kapitel noch auf Funktionstest eingegangen, die die einzelnen Anforderungen an das Framework aufgegriffen haben und durch die die volle Funktionsfähigkeit des Frameworks geprüft werden konnte.

6.2 Bewertung

Die Funktionalitäten konnten durch die in Kapitel 5 genannten Funktionstest nachgewiesen werden. Die Funktionalität ist jedoch nur eine wichtige Aufgabe der vorliegenden Arbeit gewesen. Letztlich wird das Framework für den Entwickler und nicht für Anwender geschrieben, was wiederum bedeutet, dass neben den Funktionalitäten natürlich auch die Einsatzfähigkeit und Erweiterbarkeit des Frameworks zentrale Anforderungen sind. Daher sind es die Schritte der Installation und der Einrichtung des Frameworks die eine große Bedeutung haben. Hierbei hat sich gezeigt, dass an dieser Stelle noch Nachholbedarf herrscht. Grade bei den Pfadangaben für Bilder muss noch eine Nachbesserung erfolgen, da sich diese von Server zu Server anders verhalten. Im Zusammenhang mit den Pfadangaben steht natürlich insgesamt das Problem der Portierbarkeit des Frameworks an einer zentralen Stelle. Genau an dieser muss weiterhin gearbeitet werden um das Framework besser und schneller einsatzfähig zu bekommen. Ein weiterer wichtiger Punkt in

dieser Betrachtung liegt beim Thema Sicherheit. Leider wurden nicht alle Aspekte der Sicherheit des Frameworks betrachtet und bearbeitet werden. Letztlich ist es nur eine Auswahl von Angriffsarten (siehe Kapitel 4.12) vor denen das Produkt geschützt ist. Diese sind jedoch die Angriffstechniken die zur Zeit am häufigsten Anwendung finden.Letztlich gibt es meist Probleme, die erst im Entwicklungsprozess und nicht schon im Planungsprozess auftreten. Von daher sind teilweise Funktionskonstrukte entstanden die bei vorheriger Berücksichtigung effektiver hätten entwickelt werden können. Es gilt also eine effektivere und saubere Planung und Programmierung zu forcieren, um dem Entwickler den Einsatz des Frameworks zu erleichtern.

Grundlegend kann man sagen, dass die Aufgabenstellung mit Erfolg umgesetzt wurde, wobei natürlich letztlich für ein gutes Framework noch ein wenig Arbeit zu verrichten ist.

6.3 Ausblick

Nach der Fertigstellung dieses Projektes gibt es, wie meistens eigentlich, an einigen Stellen noch Verbesserungsbedarf. Aus der Sicht des Frameworks gibt es zur Zeit vor allem bei der Installation und beim Aufbau einen Bedarf an Verbesserung beziehungsweise Erweiterung.
Zur Zeit ist die Installation des Frameworks noch in zu viele Einzelschritte und Vorbedingungen gegliedert (siehe dazu Anhang B). Diesen Schritt könnte man zu einer 1-Klick-Lösung weiterentwickeln. Dazu muss letztlich lediglich das Verzeichnis mit dem Framework in den Zielordner kopiert werden und dann eine Installationsdatei aufgerufen werden. Die Installation fragt dann nach den wichtigen Systemdaten wie zum Beispiel Datenbankverbindung sowie Passwort und richtet dann mit diesen Daten die Datenbank sowie das Framework auf dem System ein. Letztlich muss vom Anwender des Frameworks nur noch auf den Server zugegriffen und dann anhand des Frameworks die Anwendung erstellt werden. Diese Installation setzt natürlich voraus, dass eine entsprechende Routine zum kopieren der Daten und zum Einrichten der Datenbank vorhanden ist, was jedoch zum aktuellen Stand des Projektes leicht hinzugefügt werden kann.
Ein nächster Schritt in der Weiterentwicklung, auch im Zusammenhang mit der Installation des Frameworks, ist das exportieren der Daten aus der Datenbank anhand eines Befehls oder Aufrufs des Administrators. Hierbei geht es vor allem um die Datenbankstruktur und, da es sich ja um ein Framework für ein Communitynetzwerk handelt, um die Daten der Mitglieder. Da diese wiederum sensible Informationen enthalten können, ist es in diesem Fall wichtig, auf die Sicherheit der exportieren Daten zu achten und diese gegebenenfalls in einer verschlüsselten Datei zu speichern. Diese Datei wiederum kann dann für die Einrichtung der Datenbank verwendet werden.

Die Weiterentwicklung des Frameworks kann auch dahingehend vorangetrieben werden, dass der Administrator der Anwendung mehr Möglichkeiten zur Verwaltung der Daten und Mitglieder bekommen kann. Hierbei kann zum Beispiel eine Übersicht über alle Mitglieder gemacht werden mit verschieden Möglichkeiten zur Verwaltung dieser Mitglieder (z.B.: Löschen, Mahnen, Status

ändern usw.).

Das Frameworks selbst kann hinsichtlich der Gliederung und des Aufbaus noch Verbesserungen erfahren. Zwar wurde das Framework von der Architektur her so geplant, dass Entwickler dieses Framework erweitern können, doch ist dazu meist viel Einarbeitungszeit nötig. Da nun aber eben die Erweiterbarkeit und die Flexibilität wichtige Punkte bei Planung sind ist es notwendig dahingehend weitere Verbesserungen an zu streben. So könnten die unterschiedlichen Controller sowie die Views besser gegliedert werden, um somit eine bessere Modularisierung zu erreichen und dem Administrator beziehungsweise Entwickler eine bessere Übersicht zu geben.

Anlage A: Begriffserklärung

Ajax	**A**synchronous **J**avaScript **a**nd **X**ML, eine Technik die mit Hilfe des XML-Http-Requestobjektes Clientanfragen an den Server asynchron übermitteln kann
Apache	spezieller Server der Apache Software Foundation; u.a. Grundlage für die Programmiersprache PHP
Browser	Programm zum Betrachten von Webseiten und -anwendungen (z.B.:Internet Explorer, Firefox, Opera)
Client	Programm, welches vom Benutzer (=Klienten) gesteuert wird um Informationen oder Daten zum Server zu übermitteln
Controller	Schicht aus dem MVC-Konzept, verwaltet die Präsentation der Daten und stellt die Daten dem Model zur Ver- bzw. Bearbeitung zur Verfügung
CSS	**C**ascading **S**tyle **S**heets, eine HTML-Ergänzungssprache um HTML-Elemente formatieren und positionieren zu können [Selfthtml e.V.]
DOM	**D**ocument **O**bject **M**odel, Modell zum Zugriff auf Elemente von HTML oder XML
Eclipse	Entwicklungsumgebung für Programmiersprache
Framework	Funktionsgrundgerüst, das als Grundlage für die Erstellung von Anwendungen dient
GIF	**G**raphics **I**nterchange **F**ormat, Bildformat um Bilder mit geringer Farbtiefe, bis zu 256 Farben pro Bild, (fast) verlustfrei komprimieren zu können
HTML	**H**yper**t**ext **M**arkup **L**anguage, Textauszeichnungssprache zur Darstellung von Webseiten
HTTP	**H**yper**t**ext **T**ransfer **P**rotocol, ein Protokoll um vorwiegend Webseitendaten vom Webserver an den Browser zu übermitteln
Integer	Datentyp für ganzzahlige Werte
Javascript	von Netscape geschaffene Programmier- bzw. Skriptsprache um HTML-Elemente dynamisch zu verändern
JPEG	**J**oint **P**hotographic **E**xperts **G**roup, Bildformat nach der Norm ISO/IEC 10918-1
jQuery	ein Javascript-Framework zur vereinfachten DOM-Manipulation
Model	Schicht aus dem MVC-Konzept, enthält vorwiegend die Logik
MVC	Entwurfsmuster in der Softwarearchitektur mit den drei Schichten **M**odel, **V**iew und **C**ontroller
MySQL	Relationales Datenbanksystem mit Client/Server-System [Kofler, S.37]
PHP	**P**HP **H**ypertext **P**reprocessor, dynamische Programmiersprache [Bergmann,S.3]

phpMyAdmin	eine graphische Benutzeroberfläche um MySQL-Datenbanken zu verwalten
PNG	**P**ortable **N**etwork **G**raphics, Bildformat um Rastergrafiken verlustfrei komprimieren zu können
request	Anfrage eines Client an den Server
response	Antwort eines Servers an den Client
Router	spezieller Controller der benötigt wird um Anfragen an die dafür bestimmten Controller weiter zu leiten
Server	Programm, an das der Client die Daten sendet; Der Server hat die Möglichkeit über gewisse Dienste die er anbietet die Daten zu verarbeiten und mit dem Client zu kommunizieren
Session	zu deutsch Sitzung, dient zur Identifizierung von Benutzern und temporären Speicherung von Sitzungsdaten
Smarty	eine Template-Engine
SMTP-Server	**S**imple **M**ail **T**ransfer **P**rotocol-Server, ein spezieller Server über den durch das SMT-Protokoll E-Mails versendet werden können
Template-Engine	spezielle Programmiersprache um dynamische Daten in statischen Templates darzustellen (z.B.: Smarty, PHP-Template-Engine)
URL	**U**niform **R**esource **L**ocator ist eine eindeutige Bezeichnung einer Ressource im Internet (Bsp: http://www.beispiel.de)
UML	**U**nified **M**odelling **L**anguage, Sprache zur Modellierung/Beschreibung von Software und deren Bestandteilen (siehe auch *4.2.1 UML-Klassendiagramm*)
timestamp	Datentyp für einen Zeitstempel (in PHP: die Sekunden die seit 01.01.1970 vergangen sind)
Varchar	Datentyp für alphanumerische Zeichen
View	Schicht aus dem MVC-Konzept, wird für die Darstellung der Daten benötigt
XML	E**x**tensible **M**arkup **L**anguage, Metasprache zur Beschreibung von Daten [Erlenkötter, S.11]

Anlage B: Anleitung zur Installation und zum Gebrauch des entwickelten Frameworks

Installation:

Die Installation des Frameworks teilt sich in zwei Schritte. Als erstes müssen die Frameworkdateien auf den Zielrechner (Server) kopiert werden. Dabei ist zu beachten, dass der Inhalt des Ordners in dem sich die index-Datei (index.php) befindet, dahin kopiert, wo letztlich der Webserver darauf zugreifen kann. Meistens ist dies der Ordner mit dem Namen *httpdocs* auf dem Webserver. Dadurch ist gewährleistet, dass beim Aufruf der Webadresse auch direkt das Framework angesprochen wird. Sollten die Dateien des Frameworks nicht dahin kopiert werden, muss entweder der Pfad der URL angepasst werden oder das Framework und damit letztlich die Anwendung kann nicht gestartet und verwendet werden.

Der zweite Schritt der Installation ist das Einrichten der Datenbank. Dazu muss der Entwickler über die MySQL-Datenbankbenutzerschittstelle phpMyAdmin zum Beispiel den mitgelieferten SQL-Dump in den MySQL-Server importieren. Voraussetzung dafür ist natürlich, dass dem Entwickler eine Datenbank zu Verfügung steht und er die Zugangsdaten für diese kennt.

Gebrauch und Konfiguration:

Um das Framework zur Erstellung einer Anwendung zu nutzen muss dieses erst einmal konfiguriert werden. Dazu muss der Entwickler im Ordner *config* die Datei config_single_by.php in einem Editor öffnen und dort, im mit *1.1 Datenbankanbindung* gekennzeichneten Bereich, die Daten für den Zugang zur Datenbank eintragen. Um die Sprache der Modulkennzeichnung zu ändern, muss in der Konfigurationsdatei, im mit *6.1 Modulsprachdatei*, die Sprache auskommentiert werden, die nicht benötigt wird. Standardmäßig ist die Sprache deutsch voreingestellt.

Um nun aus dem Framework heraus eine Anwendung zu erstellen, ist es notwendig die Templatedateien im Unterordner *template* im Ordner *design* zu bearbeiten. Hierbei wird lediglich HTML-, CSS- und gegebenenfalls auch Javascriptwissen benötigt. Darüber können nun Oberflächen erstellt werden die bereits voll funktionsfähig sind.

Damit Entwickler das Framework mit eigenen Funktionalitäten ausstatten können müssen Sie folgenden Ablauf realisieren:

Im Ordner *classes* kann im jeweiligen Unterordner mit dem Namen einer neuen Klasse eine Datei mit der neuen Klasse hinterlegt werden. Damit die neue Klasse dann als Objekt im Framework zur Verfügung steht, muss diese, in der Datei mit dem Namen *include.php* im Ordner *skripte*, instanziert werden. Nun können im Ordner *skripte* neue Controller-Dateien angelegt werden, die

dann wiederum neue Templates aufrufen und die Methoden des neuen oder der bereits bestehenden Objekte verwenden können. Die neuen Templates werden dann im Unterordner *template* im Ordner *design* abgelegt.

Literaturverzeichnis

- Bachfeld, Daniel <dab@heise.de>: Schutz vor Attacken durch Cross-Site-Request-Forgery ausgehebelt. URL: <http://www.heise.de/ix/meldung/Schutz-vor-Attacken-durch-Cross-Site-Request-Forgery-ausgehebelt-6769.html> erstellt am 21.07.2009
- Bergmann, Sebastian: Professionelle Softwareentwicklung mit PHP 5 - 1. Aufl. - Heidelberg:dpunkt.verlag, 2005
- Bundesamt für Sicherheit in der Informationstechnik <bsi@bsi.bund.de>: Sicherheit von Webanwendungen. URL: <https://www.bsi.bund.de/cae/servlet/contentblob/476464/publicationFile/30642/WebSec_pdf.pdf>, erstellt im August 2006
- Codecasters GmbH <adj@codecasters.com>: Session Hijacking. URL: <http://www.linux-konkret.de/security/angriffsmethoden-session-hijacking-cookies-url.html>, verfügbar am 15.09.2009
- Eder, Norbert <csharp@gmx.at>: Einführung in die Framework-Entwicklung URL:<http://blog.norberteder.com/content/files/tutorials/frameworks/2007_01_18_Designrichtlinien_Frameworks_1.pdf>, erstellt am 21.01.2007
- Eichborn, Joshua <josh@bluga.net>: phpDocumentor Documentation Choices URL:<http://manual.phpdoc.org/>, erstellt am 05.09.2009
- Erack Netzwork <>: MySQL - SQL Injection Prevention URL:<http://www.tizag.com/mysqlTutorial/mysql-php-sql-injection.php>, verfügbar am 15.07.2009
- Erlenkötter, Helmut: XML - Reinbek bei Hamburg: Rowohlt Taschenbuch Verlag GmbH, 2003
- Jeckle, Mario <mario@jeckle.de> : Use-Case-Diagramm URL:<http://www.jeckle.de/uml-glasklar/UseCaseDiagramm.pdf>, erstellt am 7.6.2004
- Kofler, Michael: MySQL 5 - 3. Aufl. - München: ADDISON-WESLEY, 2005
- Krause, Jörg: Programmieren lernen in PHP 5 - München:Carl Hanser Verlag, 2004
- Niemann, Alexander: Objektorientierte Programmierung in Java - 4. Aufl. - Bonn:verlag moderne industrie Buch AG & Co. KG, 2004
- PHP Documentation Group <doc-license@lists.php.net >:PHP-Handbuch URL:< http://de.php.net/manual/de/ >, verfügbar am 14.08.2009
- Resig, John <jeresig@gmail.com >; jQuery Documentation URL: <http://docs.jquery.com/Main_Page>, verfügbar am 23.05.2009
- Selfhtml e.V. <projekt@selfhtml.org>: Stylesheets (CSS) URL:<http://de.selfhtml.org/css/>, verfügbar am 08.10.2009

- Singer, Leif <leif@singer.sh>: Model-View-Controller Seminar Software-Entwurf. URL: <http://se.uni-hannover.de/documents/ws2004_seminar_software_entwurf/04_mvc.pdf>, verfügbar am 20.08.2009
- Spolwig, Siegfried < siegfried@spolwig.de >:Klassendiagramme in UML. URL:< http://www.schule.de/schulen/oszhdl/gymnasium/faecher/informatik/ooa-ood/uml.htm>, erstellt am 23.03.2006
- Wenz Christian: Ajax schnell + kompakt - 2. Aufl. - Paderborn:entwickler.press, 2007
- Wetsch, Bruno(1) <bruno@modrewrite.de>: Die Syntax von mod_rewrite. URL:<http://www.modrewrite.de/mod_rewrite.syntax.phtml>, verfügbar am 11.08.2009
- Wetsch, Bruno(2) <bruno@modrewrite.de>: Einsatzgebiete und Anwendungen von mod_rewrite. URL:<http://www.modrewrite.de/mod_rewrite.usage.phtml>, verfügbar am 11.08.2009
- Wikimedia Foundation Inc.<info@wikimedia.org>: Template Engine. URL: <http://de.wikipedia.org/wiki/Template_Engine#Vorteile>, verfügbar am 05.08.2009
- Winkler, Jan: JavaScript Das Praxisbuch - Poing: Franzis' Verlag, 2003
- Zimmmer, Frank <zimmer@htwm.de>: Folien zur modularen Programmierung unter PHP (Webprogrammierung Teil 1). URL: <https://www.htwm.de/mmlab/intranet/doc_download.php?file=080619101652.pdf&title=Modulare_Programmierung.pdf>, verfügbar am 03.05.2009

Erklärung zur selbständigen Anfertigung der Arbeit

Erklärung

Ich erkläre, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Dresden, 12.10.2009 _____